THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 1 Capitole

Présentée et soutenue par

Munyque MITTELMANN

Le 1 septembre 2022

Logiques pour la Représentation et la Conception d'Enchères

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Informatique et Télécommunications**

Unité de recherche :
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Laurent PERRUSSEL

Jury

M. Michael THIELSCHER, Rapporteur
Mme Natasha ALECHINA, Rapporteure
M. Giuseppe PERELLI, Examinateur
M. François SCHWARZENTRUBER, Examinateur
M. Laurent PERRUSSEL, Directeur de thèse
Mme Leila AMGOUD, Présidente

Abstract

An auction is a popular mechanism that aggregates participants' bids into a social decision, usually expressed in terms of allocations and payments. Automated agents are widely used in auction-based markets, but they are usually designed to act on a specific context. Those agents cannot switch between different kinds of markets. For doing so, they should be able to "understand" the auction rules and reason about their own valuations and about other players' private information valuations. This limitation inspires the development of a lightweight logic-based language for representing the rules of an auction market, which will then allow automated general players to reason strategically in different environments.

Another important problem is the design of new auctions and, more generally, mechanisms. The challenge here is to aggregate individual preferences, while choosing a socially desirable outcome and reaching an equilibrium despite the fact that agents can lie about their preferences. Although logic-based languages have been widely considered in the context of Multi-Agent Systems (MAS), the use of formal methods and strategic reasoning for Automated Mechanism Design (AMD) has not been much explored yet.

This thesis investigates an application of logics and strategic reasoning for Game Theory and MAS. In particular, we propose the use of formal methods for the specification, design and evaluation of mechanisms, with focus on auctions. This thesis addresses such challenges by introducing logic-based approaches for representing and designing auction-based markets with strategic players.

Firstly, for providing a foundation for general and automated auction playing in MAS, we propose a framework for representing auctions, denoted Auction Description Language (ADL). ADL addresses important dimensions of auction-based markets and is general enough to represent most auction settings. We illustrate the generality of ADL by modelling a number of representative auctions. We extend ADL with knowledge operators and an action modality for characterizing bounded rational behavior of bidders when reasoning about the effect of actions and other agents' rationality.

Second, we propose a novel approach for reasoning and designing new auctions (and, in general, preference aggregation mechanisms) based on formal methods. Such approach for AMD aims to automatically generate mechanisms from their specifications and verify them in relation to target economical properties. For verifying mechanisms, we propose a new variant of Strategy Logic (SL) with quantitative semantics and epistemic operators. We demonstrate how it can express key concepts from Economic Theory, including Nash equilibrium, strategyproofness and individual rationality, which are at first importance when designing new auctions and when agents need to reason about their properties. We also introduce a quantitative semantics for SL with natural strategies and imperfect information which enables reasoning about mechanisms based on the complexity of strategies. The

analysis of mechanisms and their strategies boils-down to model checking formulas from those SL variants. Finally, we offer a novel perspective on the design of mechanisms by rephrasing the AMD problem in terms of synthesis from specifications in Quantitative SL. This approach enables automatically generating optimal mechanisms from a quantitative logical specification, which may include not only game rules but also requirements over the strategic behavior of participants and quality of the outcome.

Resumé

Une enchère est un mécanisme compétitif qui permet d'allouer un ensemble de ressources auprès d'un ensemble d'agents. Ce mécanisme agrège les offres effectuées par les participants à l'enchère dans le but de produire une décision sociale caractérisée en termes d'allocations et de paiements. Les agents automatisés sont largement utilisés sur les marchés basés sur les enchères, mais ils sont généralement conçus pour agir dans un contexte spécifique. Pour passer d'un type de marché à un autre, les agents doivent être capables de "comprendre" les règles de l'enchère et de raisonner sur leurs valuations ainsi que sur les valeurs privées des autres joueurs. De fait, il est nécessaire de définir un langage simple permettant de représenter les règles d'un marché aux enchères, qui permettra ensuite à des joueurs généraux automatiques de raisonner stratégiquement dans différents environnements. Cet aspect stratégique, notion centrale de la théorie des jeux et des systèmes multi-agents, est de première importance dans la caractérisation des mécanismes d'enchères.

Un problème de première importance concerne la conception de nouvelles enchères, ou plus généralement, de nouveaux mécanismes. En effet, un des principaux objectifs consiste à agréger les offres individuelles tout en garantissant un résultat socialement souhaitable. La dimension stratégique est donc au cœur de la conception de mécanismes. Alors que les langages logiques ont été largement considérés dans le contexte des Systèmes Multi-Agents (SMA), l'utilisation de méthodes formelles et de raisonnement stratégique pour la Conception Automatique de Mécanismes a été à peine étudiée.

Cette thèse explore l'application des logiques pour la description et la conception de mécanismes d'enchères. Ces derniers placent la dimension stratégique en leur cœur et nous proposons l'utilisation de méthodes formelles pour la spécification, la conception et l'évaluation de mécanismes intégrant cette dimension.

Dans un premier temps, afin de fournir une fondation pour les joueurs d'enchères généraux et automatisés dans les SMA, nous proposons un formalisme pour représenter les enchères, appelé Auction Description Language (ADL). ADL traite des dimensions importantes des marchés basés sur des enchères et est suffisamment général pour représenter la plupart des contextes d'enchères. Nous montrons qu'en enrichissant ADL avec un opérateur de connaissance et une modalité d'action pour caractériser le comportement rationnel limité des enchérisseurs, les agents enchérisseurs peuvent raisonner sur l'effet des actions ainsi que sur la rationalité des autres agents.

Dans un second temps, nous proposons une nouvelle approche pour le raisonnement et la conception de nouvelles enchères basée sur des méthodes formelles. Cette approche vise à générer des mécanismes à partir de leurs spécifications et à les vérifier par rapport à des propriétés économiques objectives. Nous proposons une nouvelle variante de Strategy Logic (SL) avec une sémantique quantitative et des opérateurs épistémiques. Nous montrons comment elle permet d'exprimer des

concepts essentiels de la théorie de l'économie, notamment l'équilibre de Nash et la manipulation stratégique, qui sont de première importance lors de la conception de nouvelles enchères et lorsque les agents doivent raisonner sur leurs propriétés. Nous introduisons aussi SL avec des stratégies naturelles, qui permet de raisonner sur les mécanismes en fonction de la complexité des stratégies. L'analyse des mécanismes et des stratégies se résume donc à la vérification de formule en SL dans des modèles représentant des enchères. Enfin nous proposons la reformulation du problème de la conception de mécanismes en termes de synthèse de spécifications logiques. Cette approche permet de générer automatiquement des mécanismes optimaux à partir d'une spécification, qui peut inclure non seulement les règles du jeu mais aussi des exigences sur le comportement stratégique des participants.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my thesis supervisor, Laurent Perrussel, for teaching me how to become a researcher, and also for his constant support and encouragement. This thesis would not be possible without his kind help and guidance.

I would like to thank all the committee members for being part of the jury for my Ph.D. defense: Natasha Alechina, Leila Amgoud, Giuseppe Perelli, Laurent Perrussel, François Schwarzentruber, and Michael Thielscher. In particular, I want to thank the *rapporteurs* Natasha Alechina and Michael Thielscher for having accepted reading and reviewing my thesis.

During my Ph.D., I had the opportunity to meet, talk and work with incredibly talented researchers. I am grateful for everything I learned while working with Francesco Bellardinelli, Sylvain Bouveret, Andreas Herzig, Wojtek Jamroga, and Vadim Malvone. I would also like to thank Jonathan Ben-Naim, Giuseppe de Giacomo, Umberto Grandi, Jérôme Lang, Emiliano Lorini, and Giuseppe Perelli for their fruitful comments and advice. I extend my gratitude to the AGAPE and LILAC teams, which provided many interesting seminars and discussions.

I am extremely grateful to Aniello Murano for making me feel welcome in his laboratory and for the joint collaboration throughout these years. His energy and excitement for research (and Napoli!) are inspiring and contagious. I would like to extend my sincere thanks to Bastien Maubert, for helping me to improve my mathematical skills and for being great to work with.

My sincere thanks also go to my friends and fellow Ph.D. students, in special to Rachael Colley for the teamwork in organizing our classes and for sporadically proofreading my writing.

I would also like to thank the ANR project AGAPE for having supported my Ph.D. research and to the University of Toulouse for providing me with a wonderful learning environment.

Many thanks to my parents, Mauristela and Euzébio, for all of the sacrifices that you've made on my behalf. Thanks to my wonderful siblings, Maytê and Gustavo, from whom I regret being far away during these last years. I thank my dearest friend Késsi: I was really lucky when you decided to randomly interrupt my reading to chat a decade ago. Last but not least, thank you Tárcio for your sincere patience, caring, and for always believing in me even when I felt discouraged.

Undertaking this Ph.D. has been a life-changing experience for me. I have grown up not only professionally but also personally for which I truly thank you all for the support and motivation that led me here.

Contents

Abbreviations

ADL Auction Description Language

ADLK Epistemic Auction Description Language

AMD Automated Mechanism Design

ATL Alternating time Temporal Logic

BNF Backus-Naur Form

DSE Dominant Strategy Equilibria

ET Epistemic State Transition

GDL Game Description Language

GGP General Game Playing

GSP Generalized Second Price

MAS Multi-Agent Systems

NatSL$[\mathcal{F}]$ Natural Quantitative Strategy Logic

NE Nash Equilibria

SCF Social Choice Function

SL Strategy Logic

SL$[\mathcal{F}]$ Quantitative Strategy Logic

SLK$[\mathcal{F}]$ Quantitative Epistemic Strategy Logic

ST State Transition

TBBL Tree-Based Bidding Language

VCG Vickrey–Clarke–Groves

wCGS Weighted Concurrent Game Structure

wCGSii Weighted Concurrent Game Structure with Imperfect Information

wCGSl Weighted Concurrent Game Structure with Imperfect Information and Legality

WE Weighted Epistemic

Chapter 1

Introduction

An auction is a popular mechanism that aggregates participants' bids into a social decision, usually expressed in terms of allocations and payments. The goal of an auction is to determinate the winners and their payments, when the precise price of the items considered in the market is unknown or unclear. Classical examples of such items include art pieces, jewelry and antiques. According to Chevaleyre et al. (2006), auctions are characterized by having a centralized allocation procedure that specify trades of items and by considering monetary transfers (that is, payments). Being centralized means a single entity decides on the final allocation of resources amongst agents. The central entity in an auction is the auctioneer and the reporting of preferences takes the form of bidding. Agents are not required to reveal their true preferences during bidding, but they may bid whatever they believe to fit their own interests.

The importance of Auction Theory is many-fold. From a practical point of view, they are used for negotiating a large number of goods and services, property, and financial instruments (Klemperer, 1999). Moreover, new auctions-based markets are being designed for considering novel trading and allocation problems, including, for instance, e-commerce advertising (Liu et al., 2021), smart grids (Li et al., 2020), smart contracts on blockchain (Galal and Youssef, 2019) and renewable energy technologies (Rubio-Domingo and Linares, 2021). From an empirical perspective, auctions provide a valuable testing-ground for Economic Theory. The interest in experimental work on auction has been prominent in the literature (Klemperer, 1999). Finally, auctions are also relevant for theoretical reasons. For instance, they have been important in developing the understanding of price formation. In fact, as pointed by Krishna (2009), auctions are used precisely because the seller is unsure about the values that bidders attach to the object being sold and the uncertainty regarding values is an inherent feature of auctions. Additionally, auction-theoretic models and techniques also apply to mechanisms without monetary transfers and can help develop models of oligopolistic pricing (Klemperer, 1999).

Automated agents are widely used in auction-based markets but software agents are usually designed to act on a specific context (see, for instance, auto-bidding frameworks in online advertising (Wen et al., 2022)). Those agents cannot switch between different kinds of markets. For doing so, they should be able to "understand" the auction rules and reason about their own valuations and also about other players' private information valuations. This limitation inspires the development of a lightweight logic-based language for representing the rules of an auction market, which will then allow automated general players to reason strategically in different

environments.

More than describing, the design and evaluation of new auctions (and, more generally, *mechanisms*) is a central problem in multiagent settings (Conitzer and Sandholm, 2002). In such setting, we need to be able to aggregate individual preferences, which are conflicting when agents are self-interested. More importantly, the mechanism should choose a socially desirable outcome and reach an equilibrium despite the fact that agents can lie about their preferences (Asselin et al., 2006). Automating the process of creating and evaluating mechanisms has numerous advantages, as stated by Sandholm (2003):

- It can be used in settings beyond the classes of problems that have been already studied in manual mechanism design, which is a clear link with the general dimension of auctions;
- It can yield better mechanisms (in relation to the quality of the outcome and/or avoiding strategic manipulation) because it can be constructed over the particulars of the setting. This is the case, for instance, when designing auctions for selling airport landing slots or rights to transmit signals over specific bands (spectrum auctions);
- It shifts the burden of designing a mechanism from humans to a machine, which may be a difficult and time-consuming task when considering complex aggregation problems and strategic solution concepts.

Although logic-based languages have been widely used for verification (Clarke et al., 2018) and synthesis (David and Kroening, 2017) of Multi-Agent Systems (MAS), the use of formal methods for reasoning about auctions under strategic behavior as well as automated mechanism design has not been much explored yet. An advantage in adopting such perspective lies in the high expressivity and generality of logics for strategic reasoning (Pauly and Wooldridge, 2003). Moreover, by relying on precise semantics, formal methods provide tools for rigorously analysing the correctness of systems, which is important to improve trust in mechanisms generated by machines. The problem of formally reasoning about mechanisms is, however, nontrivial: it requires to consider quantitative information (*e.g.*, utilities and payments), private information about the participant's preferences, and complex strategic concepts (such as strategy dominance and equilibria).

This thesis addresses such challenges by introducing logic-based approaches for representing and designing auction-based markets with strategic players. Our motivation is two fold: first, we aim to provide a foundation for general and automated auction playing in MAS, by establishing a logical framework to create a good balance between expressive power and computational efficiency. Second, we propose a novel approach based on formal methods for (i) reasoning about auctions under strategic behavior and (ii) Automated Mechanism Design. Such approach aims to automatically generate auctions from their specifications and verify them in relation to target economical properties.

1.1 Background and Context

This section gives an overview of the context and the main related works to which this thesis is related.

1.1.1 Representation of Auctions

There are numerous variants of auctions depending on the parameters considered, including the number of distinct items and copies as well as the number of sellers and buyers (Klemperer, 1999; Krishna, 2009). For a fixed set of parameters, the protocol, *i.e.*, the bidding, payment and allocation rules, may also differ. Building intelligent agents that can switch between different auctions and process their rules is a key issue for building automated auction-based marketplaces. In this scenario, the auctioneer should at first allow participants to express their preferences and second describe the rules governing the market.

A number of bidding languages have been proposed for conveniently expressing preferences, specially when considering combinatorial settings (Nisan, 2004). Such languages focus on compactly representing the space of possible bids over combinations of items. In this work we are interested on the representation of auction rules. For a comparative overview of classical bidding languages (*e.g.*, OR and XOR languages), the reader may refer to Nisan (2000).

In relation to the formal representation of auction rules, we recall the descriptive auction language (Rolli et al., 2006), which allows the specification of auctions by allowing players to bid using the XOR language. Rule-based approaches have also being used for representing single-dimensional auctions (Lochner and Wellman, 2004) and negotiation protocols (Bădică et al., 2006). Similarly, negotiation protocols have being handled with meta-languages (Hudert et al., 2009b), the Extensible Markup Language (Hudert et al., 2009a) and rule-markup languages (Dobriceanu et al., 2007).

Since the above languages lack a precise semantics, Wooldridge and Parsons (2000a, 2000b) motivate and compare the use of different logical languages for specifying negotiation protocols. In the context of General Game Playing (GGP), the Game Description Language (GDL) was designed for specifying game rules while maintaining a tractable complexity (Genesereth and Thielscher, 2014). The use of languages inspired on GDL for describing market-based protocols have been studied in the context of negotiation (Jonge and Zhang, 2016; de Jonge and Zhang, 2021) and single-item markets (Thielscher and Zhang, 2010). Such approaches lack a clear link between the language, the mechanism formalization and the agents' preferences, which is a key aspect for enabling reasoning about auctions.

1.1.2 Automated Mechanism Design

Designing an auction in such manner to ensure features of the outcome alongside with a desirable behavior of the participants is a key challenge in Economics. In fact, this problem is known as *Mechanism Design*: the formulation of game-like

systems whose equilibria satisfy some desired properties, usually expressed in terms of incentive, utility or social welfare. Traditionally, mechanisms have been formulated and verified by human specialists, who use their knowledge and experience for defining the game rules. Creating and verifying mechanisms which will be played by strategic agents can be a very difficult and time-consuming task.

Sandholm (2003) introduced Automated Mechanism Design (AMD), whose goal is to automatically create mechanisms for solving a specific preference aggregation problem. AMD is usually handled as an optimization and domain-oriented problem. Most solutions used on the literature are based on machine learning, which include, for instance, neural networks (Shen et al., 2019; Dütting et al., 2019) and statistical techniques (Narasimhan et al., 2016). A number of works explore computed-aided verification of auctions (Caminati et al., 2015; Barthe et al., 2016; Kerber et al., 2016), where the process is assisted by a reasoner. In the context of fully-automatic verification, Pauly and Wooldridge (2003) and Wooldridge et al. (2007) advocate the use of Alternating-time Temporal Logic (ATL) (Alur et al., 2002) to reason about mechanisms. The main limitation in these works is the purely qualitative setting and the impossibility of expressing key strategic concepts such as dominance in the logic.

1.1.3 Logics for Strategic Reasoning

This thesis is also related to the long-established logical approach to systems verification (Clarke et al., 2018) and synthesis (David and Kroening, 2017). In the recent years much progress has been made in the field of logics for strategic reasoning. Pioneering work includes the Coalition Logic (Pauly, 2002) and the aforementioned ATL (Alur et al., 2002). These logics use coalition modalities to specify strategic abilities of groups of agents, an important notion in Mechanism Design. A logic for reasoning about composite strategies in turn-based games is introduced by Ramanujam and Simon (2008), where strategies are treated as programs that are combined by connectives from propositional dynamic logic. The Strategy Logic (SL) (Chatterjee et al., 2010; Mogavero et al., 2014) subsumes ATL and considers explicit manipulation of strategies. This feature allows expressing important concepts in games, such as Nash equilibria. A recent quantitative extension of SL, denoted SL[$\mathcal{F}$] (Bouyer et al., 2019), 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 (Belardinelli et al., 2020; Berthon et al., 2021; Cermák et al., 2018), which is also an important feature when modeling auctions with private valuations.

1.2 Contribution

This thesis addresses the problem of modelling and analyzing auction mechanisms for MAS using logics and strategic reasoning. The main contributions of this work

can be summarized as follows:

1. We provide a framework for representing auctions, denoted Auction Description Language (ADL). ADL addresses important dimensions of auction-based markets and is general enough to represent most auction settings. We illustrate the generality of ADL by modelling a number of representative auctions;

2. We demonstrate how ADL can be used for the automated verification of direct mechanisms and for automatically checking well-formedness of auction descriptions;

3. We extend ADL with knowledge operators and an action modality (denoted Epistemic ADL, or simply ADLK) for providing a ground for the design of general auction players and characterizing their rational behavior when reasoning about the effect of actions and other players rationality;

4. We propose a new variant of Strategy Logic with quantitative features, imperfect information and epistemic operators, that we call $\mathsf{SLK}[\mathcal{F}]$;

5. We demonstrate how $\mathsf{SLK}[\mathcal{F}]$ can express the implementation of social choice functions and be used for automatically verifying a number of important properties often required in auctions, or more generally in mechanism design. We also show how we can express properties relating agents' revenues with respect to their beliefs about other agents' preferences;

6. We show that verifying a mechanism in relation to classical properties boils-down to model checking a $\mathsf{SLK}[\mathcal{F}]$ formula and we prove it can be done in PSPACE for memoryless strategies;

7. We introduce a quantitative semantics for SL with natural strategies and imperfect information (denoted $\mathsf{NatSL}[\mathcal{F}]$) which provides a new perspective for formal verification and design of novel mechanisms based on the complexity of strategies;

8. We prove that the model-checking problem for $\mathsf{NatSL}[\mathcal{F}]$ is PSPACE-complete and that $\mathsf{NatSL}[\mathcal{F}]$ has incomparable distinguishing and expressive power to $\mathsf{SL}[\mathcal{F}]$;

9. We offer a novel perspective on the design of mechanisms by rephrasing the AMD problem in terms of synthesis from $\mathsf{SL}[\mathcal{F}]$ specifications. This approach enables automatically generating optimal mechanisms from a quantitative logical specification, which may include not only game rules but also requirements over the strategic behavior of participants and quality of the outcome;

10. We solve the synthesis problem for $\mathsf{SL}[\mathcal{F}]$ by investigating the related satisfiability problem in two cases: when the number of actions is bounded, and when agents play in turn.

1.3 Outline of Chapters

This thesis is divided in two parts. The Related Work is spread out at the first chapter of each part. Part I is focused on auction specification using logical languages. Chapter 2 details the related works on the representation of auction-based markets. We then introduce ADL for describing auction rules, and provide its semantics based on state-transition models. Next, we explore the evaluation and characterization of protocols described in ADL. Finally, we examine the complexity of model checking ADL-formulas, that is, the problem of deciding whether a formula holds in a stage of a path under a state-transition model.

Chapter 3 illustrates the generality of ADL by demonstrating its use for representing and evaluating representative auctions. We consider a simultaneous ascending auction, a Vickrey-Clarke-Groves mechanism and an iterative combinatorial exchange.

Chapter 4 extends a simplified version of ADL with epistemic operators and action modalities. We characterize agents' rationality in relation to high-order knowledge about other agents and a finite number of stages looked ahead. We illustrate the use of ADLK for reasoning about Dutch auctions and examine its model-checking complexity.

Part II investigates logical and strategic reasoning in mechanism design. In Chapter 5, we focus on the verification of mechanisms. We first propose a new variant of Strategy Logic with quantitative features, imperfect information and epistemic operators, that we call $\mathsf{SLK}[\mathcal{F}]$. We show how mechanisms can be cast as concurrent-game structures and how $\mathsf{SLK}[\mathcal{F}]$ can be used to express and verify fundamental concepts from mechanism design, including the implementation of social choice functions and a number of classical properties (*e.g.*, strategyproofness, individual rationality and efficiency), which are at first importance when automated agents need to reason about an auction-based market. Finally, we study the complexity for model checking $\mathsf{SLK}[\mathcal{F}]$-formulas.

Chapter 6 extends $\mathsf{SL}[\mathcal{F}]$ with natural strategies. We then investigate its use for reasoning about simple strategies and equilibria in repeated keyword auctions. Finally, we analyze this logic in relation to its distinguishing power, expressivity, and model checking complexity for natural strategies with and without recall.

Chapter 7 offers a novel perspective on the design of mechanisms. We propose the generation of optimal mechanisms from a quantitative logical specification. For doing so, we rephrase the AMD problem in terms of synthesis of concurrent game structures from $\mathsf{SL}[\mathcal{F}]$-formulas. To solve this synthesis problem we investigate the related satisfiability problem for $\mathsf{SL}[\mathcal{F}]$, which had not been studied so far. Finally, we illustrate the relevance of mechanism synthesis with examples based on auctions.

Chapter 8 summarizes the main results of this thesis and discusses some directions for future work. Appendix A introduces basic notation from complexity theory. Appendix B presents additional details for the satisfiability proof outlined in Chapter 7, and Appendix C lists the papers published papers during my PhD from which this thesis is based.

Part I

Representing Auctions

Chapter 2

Logic for Auction Specification

In this chapter we aim at proposing a language with clear semantics to enable the representation of auctions as well as the reasoning about its rules. Auctions may differ in several ways, from the setting considered to the availability of an expressive bidding language. According to the classification proposed by Kalagnanam and Parkes (2004), let us first recall the six core dimensions of auctions:

1. Resources: the auction may involve a single item or multiple items, with single or multiple units of each item. The type of the item may also be considered, *i.e.*, the item may be a multi-attribute commodity, that is, the item may be characterized by more attributes than just the price (such as the service time, quality, and payment terms);

2. Market structure: the auction may be distinguished based on whether it has single or multiple buyers and/or sellers. In single-sided auctions, there is either a single seller selling resources to multiple buyers (*i.e.*, forward auctions), or a single buyer sourcing resources from multiple suppliers (*i.e.*, reverse auctions). In a double auction, there are multiple suppliers selling resources to multiple buyers. Finally, an exchange is a generalization of double auctions, where the participants *trade* items;

3. Preference structure: the participants' preferences define their utilities over different outcomes (*e.g.*, marginal utility, quasi-linear utility);

4. Bid structure: it refers to the flexibility with which participants can express their preferences. The bid structure ranges from simple statements of willingness to accept a given selling price to complex bids that state prices, quantities, bundles, and logical connectives. In single-dimensional auctions, only one attribute is considered (*e.g.*, the price offered for a good). On the other hand, multi-dimensional auctions handle a number of attributes in the bid (*e.g.*, quality, delivery date) (Parsons et al., 2011);

5. Market clearing: in relation to the method for matching the supply to demand, an auction may be single-sourcing (*i.e.*, matching pairs of buyers and sellers) or multi-sourcing (*i.e.*, matching multiple sellers with a single buyer, or vice-versa);

6. Information feedback: auctions may also differ on whether they are direct (*i.e.*, one-stage protocols without information feedback) or indirect mechanisms

(*i.e.*, protocols where agents can adjust their bids in response to information feedback).

In the spirit of General Game Playing (GGP) (Genesereth and Thielscher, 2014) where games are described with the help of the Game Description Language (GDL), we propose the *Auction Description Language* (ADL). Such logical language builds upon bidding languages, and hence provides a natural way to represent a wide range of protocols, ranging from single-units auctions to iterative combinatorial exchanges (Parkes et al., 2005). As for GDL, we propose a precise semantics based on state-transition models, that gives a clear meaning to auction rules. With ADL, we can use different bidding languages for describing the set of possible actions, including the Tree-Based Bidding Language (Parkes et al., 2005), which generalizes known languages such as XOR and OR (Nisan, 2000) to combinatorial exchange. To the best of our knowledge, ADL is the first framework offering a unified perspective on an auction mechanism and it offers two benefits: (i) with this language, one can represent many kinds of auctions in a compact way and (ii) the precise state-transition semantics can be used to derive key properties.

2.1 Related Work

Our work is rooted in the key contributions on Auction Theory (Nisan, 2000, 2004; Xia et al., 2005; Krishna, 2009). All these works adopt a mechanism design perspective: they focus on designing and evaluating protocols and bidding languages. ADL has a different purpose. It is designed to represent auction protocols and to allow a modular definition of actions sets, which may be characterized by a bidding language. ADL can be used to automatically derive properties for these protocols. ADL is also a tool for mechanism design, since it is a well-suited framework for testing new auctions.

2.1.1 Auction-Based Markets

To the best of our knowledge, almost all contributions on the computational representation of auctions focus on the implementation of the winner determination problem. For instance, Baral and Uyan (2001) show how the winner determination for a combinatorial auction can be encoded in a logic program. A hybrid approach mixing linear programming and logic programming has been proposed by Lee and Lee (1997): they focus on sealed-bid auctions and show how qualitative reasoning helps to refine the optimal quantitative solutions. Giovannucci et al. (2010) explore a graphical formalism to compactly represent the winner determination problem for multi-unit combinatorial auctions. Linear Logic has also been used for modeling combinatorial auctions (Porello and Endriss, 2010). The authors explore the representation of bids and the winner determination. However, there is no temporal operator to allow reasoning on iterative auctions.

The descriptive auction language (Rolli et al., 2006) allows a formal specification of auctions. Agents can only bid through XOR combinations of items. Although the XOR language is widely used, the specification approach is more flexible when the bidding language is not fixed for all protocols. In fact, as stated by Nisan (2004), the choice of a bidding language should aim to find a balance between expressivity (*i.e.*, being able to express every preference function) and simplicity (*i.e.*, being intuitive and computationally efficient). There are numerous auction types that do not require the expressive power of the XOR language and can be implemented with simpler bidding languages. For instance, in a Dutch auction, agents may simply say whether they accept the current selling price or not.

A rule-based scripting language for representing single-dimensional auctions have been proposed by Lochner and Wellman (2004). Since it is single-dimensional, bids are composed exclusively by prices. On the other hand, the framework Multiple criteria English Reverse Auctions (Bellosta et al., 2005; Bellosta et al., 2008) characterizes bids with vectors of attributes and criteria. Their framework allows to represent English reverse auctions that differ in relation to model for aggregating bidders' preferences and the information feedback provided to participants.

2.1.2 Negotiation Protocols

A related problem to auction specification is the one of representing negotiation protocols. As noted by Meyer et al. (2004) negotiation is investigated from different perspectives, such as economics, psychology, computer science and others. Consequently, there is no agreement in relation to its definition and the distinction between negotiation and auctions may be vague. Although some authors consider auctions as types of negotiation protocols, in this work we consider them related, but rather distinct, types of allocation procedures.

As mentioned before, by auctions we refer to centralized mechanisms that specify trades of items and payments based on the bidders' reported preferences. On the other hand, we consider that in negotiation protocols, allocations emerge as the result of a sequence of local negotiation steps. That is, they describe the negotiation over resources in a distributed setting. Another important difference is that negotiation protocols for exchanging goods may not depend on monetary transfer.

Bădică et al. (2006) discuss how rule-based approaches can be used to automated negotiation. The paper focuses on experimental results for concurrent negotiation and do not address the generality of their approach. Hudert et al. (2009b) propose a framework to enable negotiations according to bilateral and multilateral protocols using a meta-language to define protocols based on a set of attributes and parameters. In a subsequent work, the authors propose a language based on Extensible Markup Language for meta-negotiating the choice of negotiation protocols and for instantiating and parameterizing the system used for conducting the chosen protocol (Hudert et al., 2009a). Similarly, rule-markup languages have also been used for the declarative representation of negotiations (Dobriceanu et al., 2007).

Due to the lack of a precise semantics, the approaches considered up to here

are too poor to enable reasoning. This limitation motivates the study of negotiation protocols with logic-based languages. In two papers, Wooldridge and Parsons (2000a,b) briefly compare the use of different languages for negotiation, including propositional logic, a language for electronic commerce and a negotiation meta-language. They consider simple protocols with multi-attribute negotiation of single goods. Even in this setting, the problems of determining if agreement has been reached and determining if a particular protocol will terminate are computationally hard. A basic logical framework for negotiation has also been proposed by Meyer et al. (2004). In the two agents setting, the work explores modes of negotiation from which an agreement over an outcome can be reached when the participants are rational, cooperative and truthful.

2.1.3 Game Description Language

The *Game Description Language* (GDL) is the official language used for the GGP competition (Genesereth and Thielscher, 2014). To incorporate imperfect information games, GDL has been extended to GDL-II (Thielscher, 2010) and GDL-III (Thielscher, 2017). GDL-II and GDL-III aim at describing the rules of an imperfect information game, but do not provide tools for reasoning about how a player infers information based on these rules. All these logics face decidability and tractability issues: their expressive power prevents them from being implemented realistically in an artificial agent. Jiang et al. (2016) propose an epistemic extension of GDL (E-GDL) to represent and reason about imperfect information games. Their language allows us to represent the rules in the imperfect information setting. A key characteristic of E-GDL is that it manages the balance between expressiveness and computational complexity of model checking (Δ_2^{P}). GDL has also been extended for defining, comparing and combining strategies in (Zhang and Thielscher, 2015a,b). This extension includes the dual connectives *prioritised disjunction* and *prioritised conjunction* for expressing preferences when combining strategies.

The use of GDL-based languages for describing market-based protocols has also been studied. Jonge and Zhang (2016) discuss the use of GDL for modeling negotiation. The main advantage is being able to apply the existing domain-independent techniques from GGP. For instance, the Monte Carlo Tree Search algorithm has been adapted for negotiations in non-zero-sum games (Jonge and Zhang, 2017). In another paper, they propose the use of GDL as a unifying language for defining general and complex negotiation domains (de Jonge and Zhang, 2021).

The Market Specification Language proposed by Thielscher and Zhang (2010) is also based on GDL. The work focuses on representing single item auctions through a set of rules and then interpreting an auction-instance with the help of a state-based semantics. However, the main limit is the lack of a clear link between the language, the mechanism formalization and the agents' preferences, which prevents the evaluation of GDL-based specifications as mechanisms.

2.2 Auction Description Language

Let us now define the Auction Description Language (ADL), which is a framework for specification of auction-based markets composed by a state-transition model and a logical language. To encode an auction, we first define its signature, which specifies the participants (the agents), the goods being traded and the propositions and variables describing each state of the auction:

Definition 2.1. An *auction signature* $\mathcal{S}$ is a tuple $(\mathrm{N}, \mathrm{G}, \mathcal{B}, \Phi, \mathrm{Y}, \mathrm{I}, \mathcal{F})$, where:

- $\mathrm{N} = \{1, \ldots, \mathsf{n}\}$ is a nonempty finite set of *agents* (or bidders);
- $\mathrm{G} = \{1, \ldots, \mathsf{m}\}$ is a finite set of good[1] types;
- $\mathcal{B}$ is a nonempty finite set of *bids* (or actions);
- Φ is a finite set of atomic propositions specifying features of a state;
- Y is a finite set of numerical variables specifying numerical features of a state;
- $\mathrm{I} \subseteq \mathbb{Z}$ is an interval of the form $[z_{\min}, z_{\max}]$ denoting the value range for any countable component of the framework, where $z_{\min} \in \mathbb{Z}$, $z_{\max} \in \mathbb{Z}$, $z_{\min} \leq 0$ and $z_{\max} \geq 0$;
- $\mathcal{F} \subseteq \{f : \mathcal{B}^{\mathsf{a}} \times \mathrm{I}^{\mathsf{b}} \to \mathrm{I} \mid \mathsf{a}, \mathsf{b} \in \mathrm{I}_{\succeq 0}\}$ is a set of state-independent functions of possibly different arities.

We assume I is equipped with a partial order[2] $\preceq$, capturing the standard less-than-or-equal relation over its elements. By convenience, we assume $0 \in \mathrm{I}$ and denote $\mathrm{I}_{\succeq 0} = [0, z_{\max}]$ and $\mathrm{I}_{\succ 0} = (0, z_{\max}]$ as the non-negative and positive parts of I, respectively. Similarly, $\mathrm{I}_{\preceq 0} = [z_{\min}, 0]$ and $\mathrm{I}_{\prec 0} = [z_{\min}, 0)$ denote the non-positive and negative parts of I.

Hereafter, we will fix an auction signature $\mathcal{S}$ and all concepts will be based on this signature, except if stated otherwise. Note that $z_{\min}$ and $z_{\max}$, in the definition of I, should be large enough to represent the total supply of goods being traded as well as the cumulative available money among agents. Throughout the rest of this chapter, we assume that is the case. Values outside I are rounded to the nearest value $z_{\min}$ or $z_{\max}$. We assume that $\mathcal{F}$ contains the functions $\mathrm{sum}(z_1, z_2) = \min(z_{\max}, z_1 + z_2)$, $\mathrm{sub}(z_1, z_2) = \max(z_{\min}, z_1 - z_2)$, $\mathrm{times}(z_1, z_2) = \min(z_{\max}, z_1 \cdot z_2)$, denoting the addition, subtraction and multiplication of two integers $z_1, z_2 \in \mathrm{I}$, respectively. . Similarly, we also assume $\mathcal{F}$ contains the functions

$$\max(z_1, z_2) = \begin{cases} z_1 & \text{if } z_2 \preceq z_1 \\ z_2 & \text{otherwise} \end{cases}$$

[1] Throughout this thesis the terms "item" and "good" are used interchangeably.

[2] A partial order is a relation that is reflexive, antisymmetric, and transitive.

and

$$\min(z_1, z_2) = \begin{cases} z_1 & \text{if } z_1 \preceq z_2 \\ z_2 & \text{otherwise} \end{cases}$$

capturing the maximum and minimum among two integers $z_1, z_2 \in \mathrm{I}$.

Given a sequence $(z_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}} \in \mathrm{I}^{\mathsf{n}}$ and a 2-ary function $f \in \mathcal{F}$ such that $f : \mathrm{I}\times\mathrm{I} \to \mathrm{I}$, we will use the following shortcut:

$$f_{\mathsf{i}\in\mathrm{N}}(z_{\mathsf{i}}) = f(z_1, f(z_2, f(..., f(z_{\mathsf{n}-1}, z_{\mathsf{n}}))))$$

A *trade* is a tuple $(\lambda_{\mathsf{i},j})_{j\in\mathrm{G},\mathsf{i}\in\mathrm{N}} \in \mathrm{I}^{\mathsf{nm}}$, where $\lambda_{\mathsf{i},j}$ denotes the number of units j being traded by agent i. A trade specifies which items each agent is selling and/or buying. A positive trade expresses how many units of a good type are purchased and a negative trade represents how many units are sold. Similarly, a positive payment denotes how much a buyer will pay and a negative payment expresses how much a seller will receive.

A tuple of objects indexed by agents in N is called a *profile*. In a profile, we may omit the index set and we write it in bold, e.g., $\boldsymbol{\lambda}$ for $(\lambda_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ and $(\boldsymbol{\lambda_j})_{j\in\mathrm{G}}$ for $(\lambda_{\mathsf{i},j})_{j\in\mathrm{G},\mathsf{i}\in\mathrm{N}}$. Given a profile $\boldsymbol{o}$, we let o_{r} be the component of agent r and $o_{-\mathsf{r}}$ be $(o_{\mathsf{i}})_{\mathsf{i}\neq\mathsf{r}}$. Let us now present the model and the language's syntax and semantics.

2.2.1 Syntax

Let $\mathcal{L}_Z$ be the set of numerical terms, with each $z \in \mathcal{L}_Z$ defined as follows:

$$z ::= \mathrm{z} \mid \mathrm{var} \mid f(\beta, \ldots, \beta, z, \ldots, z)$$

where $\beta \in \mathcal{B}$ is a bid, $\mathrm{z} \in \mathrm{I}$ is an integer, $\mathrm{var} \in \mathrm{Y}$ is a numerical variable and $f \in \mathcal{F}$ is a function[3].

The logical language for ADL is denoted by $\mathcal{L}_{\mathsf{ADL}}$ and a formula φ in $\mathcal{L}_{\mathsf{ADL}}$ is defined by the following Backus-Naur Form (BNF) grammar:

$$\varphi ::= p \mid \mathrm{initial} \mid \mathrm{terminal} \mid \mathrm{legal}_{\mathsf{i}}(\beta) \mid \mathrm{does}_{\mathsf{i}}(\beta) \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi \mid z \ \leq \ z$$

where $p \in \Phi$ is an atomic proposition, $\mathsf{i} \in \mathrm{N}$ is an agent, $\beta \in \mathcal{B}$ is a bid, and $z \in \mathcal{L}_Z$ is a numerical term.

Intuitively, initial and terminal specify the initial terminal states, resp.; $\mathrm{legal}_{\mathsf{i}}(\beta)$ asserts that agent i is allowed to take action β at the current state and $\mathrm{does}_{\mathsf{i}}(\beta)$ asserts that agent i takes action β at the current state. The formula $\bigcirc\varphi$ means "φ holds at the next state". The formula $z_1 \leq z_2$ means that the numerical term z_1 is smaller or equal to the numerical term z_2.

We use the standard abbreviations from propositional logic, such as $\varphi \vee \psi$ for $\neg(\neg\varphi \wedge \neg\psi)$, $\varphi \to \psi$ for $\neg\varphi \vee \psi$, and $\varphi \leftrightarrow \psi$ for $(\varphi \to \psi) \wedge (\psi \to \varphi)$. We take $\top$ to be an abbreviation for some fixed propositional tautology such as $p \vee \neg p$, and let

[3]Notice $f(.)$ may be 0-ary.

$\bot$ be an abbreviation of $\neg\top$. We also use abbreviations for comparison operators, such as $z_1 = z_2$ for $z_1 \leq z_2 \wedge z_2 \leq z_1$, $z_1 < z_2$ for $z_1 \leq z_2 \wedge \neg(z_1 = z_2)$, $z_1 \neq z_2$ for $\neg(z_1 = z_2)$, $z_1 \geq z_2$ for $z_2 \leq z_1$ and $z_1 > z_2$ for $z_1 \geq z_2 \wedge z_1 \neq z_2$. The extension of the comparison operators to multiple arguments is straightforward.

Assume the bids $\boldsymbol{\beta} = (\beta_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$, where $\beta_{\mathsf{i}} \in \mathcal{B}$ is a bid associated to the agent $\mathsf{i} \in \mathrm{N}$. The formula $\mathrm{does}(\boldsymbol{\beta}) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N}} \mathrm{does}_{\mathsf{i}}(\beta_{\mathsf{i}})$ represents that the agents in N perform the joint action $\boldsymbol{\beta}$.

2.2.2 Semantics

As for GDL, the semantics of ADL is based on state-transition models. This is more suitable for describing the dynamics than stable models that were initially considered for GDL and GGP (Genesereth and Thielscher, 2014). Such models allows us to represent the key aspects of an auction, at first the legal bids and the transitions among states.

Definition 2.2. A state-transition-model (ST-*model* for short) M is a tuple $(\mathrm{W}, \bar{\mathrm{w}}, \mathrm{T}, \mathrm{L}, \mathrm{U}, \pi_\Phi, \pi_\mathrm{Y})$, where:

- W is a nonempty set of *states*;
- $\bar{\mathrm{w}} \in \mathrm{W}$ is the *initial* state;
- $\mathrm{T} \subseteq \mathrm{W}$ is a set of *terminal* states;
- $\mathrm{L} \subseteq \mathrm{W} \times \mathrm{N} \times \mathcal{B}$ is a *legality* relation, describing the legal actions at each state, we let $\mathrm{L}(\mathrm{w}, \mathsf{i}) = \{\beta \in \mathcal{B} \mid (\mathrm{w}, \mathsf{i}, \beta) \in \mathrm{L}\}$ be the set of all legal actions for agent i at state w;
- $\mathrm{U} : \mathrm{W} \times \mathcal{B}^{\mathsf{n}} \to \mathrm{W}$ is an *update* function, given $\boldsymbol{d} \in \mathcal{B}^{\mathsf{n}}$, let d_{i} be the individual action for agent i in the joint action $\boldsymbol{d}$;
- $\pi_\Phi : \mathrm{W} \to 2^\Phi$ is the valuation function for the state propositions;
- $\pi_\mathrm{Y} : \mathrm{W} \times \mathrm{Y} \to \mathrm{I}$, is the valuation function for the numerical variables.

A *path* represents a run or execution of an auction protocol. Formally,

Definition 2.3. Given an ST-model $\mathrm{M} = (\mathrm{W}, \bar{\mathrm{w}}, \mathrm{T}, \mathrm{L}, \mathrm{U}, \pi_\Phi, \pi_\mathrm{Y})$, a *path* is a sequence of states and joint actions $\mathrm{w}_0 \xrightarrow{\boldsymbol{d}^1} \mathrm{w}_1 \xrightarrow{\boldsymbol{d}^2} \ldots \xrightarrow{\boldsymbol{d}^t} \mathrm{w}_t \xrightarrow{\boldsymbol{d}^{t+1}} \ldots$ such that for any $t \geq 1$: (i) $\mathrm{w}_0 = \bar{\mathrm{w}}$; (ii) $\mathrm{w}_t \neq \mathrm{w}_0$; (iii) $d^t_{\mathsf{i}} \in \mathrm{L}(\mathrm{w}_{t-1}, \mathsf{i})$ for any $\mathsf{i} \in \mathrm{N}$, (iv) $\mathrm{w}_t = \mathrm{U}(\mathrm{w}_{t-1}, \boldsymbol{d}^t)$; and (v) if $\mathrm{w}_{t-1} \in \mathrm{T}$, then $\mathrm{w}_{t-1} = \mathrm{w}_t$.

For any path δ, let $\delta[t]$ denote the t-th state of δ, $\theta(\delta, t)$ denote the joint action performed at stage t of δ, $\theta_{\mathsf{i}}(\delta, t)$ denote the action of agent i performed at stage t of δ, and $\delta[0, t]$ denote the finite prefix $\bar{\mathrm{w}} \xrightarrow{\boldsymbol{d}^1} \mathrm{w}_1 \xrightarrow{\boldsymbol{d}^2} \ldots \xrightarrow{\boldsymbol{d}^t} \mathrm{w}_t$. A path δ is *complete* if $\delta[e] \in \mathrm{T}$, for some $e > 0$. After reaching a terminal state $\delta[e]$, for any $e' > e$, $\delta[e'] = \delta[e]$. Finally, for a given model M, any state w such that there exists a complete path δ of M such that $\mathrm{w} \in \delta$ will be called a *reachable state* of M.

The semantics for ADL is given in two steps. First, we define function f_Z to compute the meaning of numerical terms $z \in \mathcal{L}_Z$ in some specific state. Next, a formula $\varphi \in \mathcal{L}_{\mathsf{ADL}}$ is interpreted with respect to a stage in a path.

Definition 2.4. Given an ST-model M, we define function $f_Z : \mathcal{L}_Z \times \mathrm{W} \to \mathrm{I}$, assigning any $z \in \mathcal{L}_Z$ and state $\mathrm{w} \in \mathrm{W}$ to a number in I:

$$f_Z(z, \mathrm{w}) = \begin{cases} \pi_{\mathrm{Y}}(\mathrm{w}, z) & \text{if } z \in \mathrm{Y} \\ f(\beta_1, \ldots, \beta_n, f_Z(z_1, \mathrm{w}), \ldots, f_Z(z_m, \mathrm{w})) & \text{if } z = f(\beta_1, \ldots, \beta_n, z_1, \ldots, z_m) \\ z & \text{if } z \in \mathrm{I} \end{cases}$$

where $n, m \in \mathrm{I}_{\succeq 0}$.

Definition 2.5. Let M be an ST-Model. Given a path δ of M, a stage t on δ and a formula $\varphi \in \mathcal{L}_{\mathsf{ADL}}$, we say φ is true (or satisfied) at t of δ under M, denoted by $\mathrm{M}, \delta, t \models \varphi$, according to the following definition:

$$\begin{array}{lll}
\mathrm{M}, \delta, t \models p & \text{iff} & p \in \pi_\Phi(\delta[t]) \\
\mathrm{M}, \delta, t \models \neg\varphi & \text{iff} & \mathrm{M}, \delta, t \not\models \varphi \\
\mathrm{M}, \delta, t \models \varphi_1 \wedge \varphi_2 & \text{iff} & \mathrm{M}, \delta, t \models \varphi_1 \text{ and } \mathrm{M}, \delta, t \models \varphi_2 \\
\mathrm{M}, \delta, t \models \text{initial} & \text{iff} & \delta[t] = \bar{\mathrm{w}} \\
\mathrm{M}, \delta, t \models \text{terminal} & \text{iff} & \delta[t] \in \mathrm{T} \\
\mathrm{M}, \delta, t \models \text{legal}_{\mathsf{i}}(\beta) & \text{iff} & \beta \in \mathrm{L}(\delta[t], \mathsf{i}) \\
\mathrm{M}, \delta, t \models \text{does}_{\mathsf{i}}(\beta) & \text{iff} & \theta_{\mathsf{i}}(\delta, t) = \beta \\
\mathrm{M}, \delta, t \models \bigcirc\varphi & \text{iff} & \mathrm{M}, \delta, t+1 \models \varphi \\
\mathrm{M}, \delta, t \models z_1 \preceq z_2 & \text{iff} & f_Z(z_1, \delta[t]) \preceq f_Z(z_2, \delta[t])
\end{array}$$

A formula φ is globally true through δ, denoted by $\mathrm{M}, \delta \models \varphi$, if $\mathrm{M}, \delta, t \models \varphi$ for any stage t of δ. A formula φ is *globally true* in an ST-model M, written $\mathrm{M} \models \varphi$, if $\mathrm{M}, \delta \models \varphi$ for all paths δ in M. Finally, let Σ be a set of formulae in $\mathcal{L}_{\mathsf{ADL}}$, then M is a *model* of Σ if $\mathrm{M} \models \varphi$ for all $\varphi \in \Sigma$.

Similar to Epistemic GDL (Jiang et al., 2021), the following propositions hold:

Proposition 2.1. *Let* M *be an ST-model, for each agent* $\mathsf{i} \in \mathrm{N}$ *and each action* $\beta \in \mathcal{B}$*,*

1. $\mathrm{M} \models \text{does}_{\mathsf{i}}(\beta) \to \neg\text{does}_{\mathsf{i}}(\beta')$*, for any* $\beta' \in \mathcal{B}$ *such that* $\beta' \neq \beta$
2. $\mathrm{M} \models \bigvee_{\beta' \in \mathcal{B}} \text{does}_{\mathsf{i}}(\beta')$
3. $\mathrm{M} \models \text{does}_{\mathsf{i}}(\beta) \to \text{legal}_{\mathsf{i}}(\beta)$
4. $\mathrm{M} \models \neg \bigcirc \text{initial}$
5. $\mathrm{M} \models \text{terminal} \wedge \varphi \to \bigcirc\varphi$*, for any* $\varphi \in \mathcal{L}_{\mathsf{ADL}}$

6. $\mathrm{M} \models \text{initial} \rightarrow \neg\text{terminal}$

Proof. Let M be an ST-model, δ be a path, $t \geq 0$ be an stage in δ, $\mathsf{i} \in \mathrm{N}$ be an agent and $\beta \in \mathcal{B}$ be an action. For Statement 1, assume $\mathrm{M}, \delta, t \models \mathrm{does}_{\mathsf{i}}(\beta)$ iff $\theta_{\mathsf{i}}(\delta, t) = \beta$. Then for any $\beta' \in \mathcal{B}$ such that $\beta' \neq \beta$, $\theta_{\mathsf{i}}(\delta, t) \neq \beta'$ and $\mathrm{M}, \delta, t \models \neg\mathrm{does}_{\mathsf{i}}(\beta')$.

Statement 2 follows from the definition of δ, since $\theta_{\mathsf{i}}(\delta, t) \in \mathrm{L}(\delta[t], \mathsf{i})$ and $\mathrm{L}(\delta[t], \mathsf{i}) \subseteq \mathcal{B}$, we have $\mathrm{M}, \delta, t \models \bigvee_{\beta' \in \mathcal{B}} \mathrm{does}_{\mathsf{i}}(\beta')$.

Let us verify Statement 3. Assume $\mathrm{M}, \delta, t \models \mathrm{does}_{\mathsf{i}}(\beta)$, then $\theta_{\mathsf{i}}(\delta, t) = \beta$ and by the definition of δ, $\beta \in \mathrm{L}(\delta[t], \mathsf{i})$ and $\mathrm{M}, \delta, t \models \mathrm{legal}_{\mathsf{i}}(\beta)$.

We consider Statement 4. By the path definition, for all $t > 0$, $\delta[t] \neq \bar{\mathrm{w}}$. Thus, $\delta[t+1] \neq \bar{\mathrm{w}}$ and $\mathrm{M}, \delta, t+1 \models \neg\text{initial}$. It follows that $\mathrm{M}, \delta, t \models \neg \bigcirc \text{initial}$.

We now verify Statement 5. Assume $\mathrm{M} \models \text{terminal} \wedge \varphi$, for some $\varphi \in \mathcal{L}_{\mathsf{ADL}}$. Then $\delta[t] \in \mathrm{T}$ and $\delta[t+1] = \delta[t]$. Thus, $\mathrm{M}, \delta, t+1 \models \varphi$ and $\mathrm{M}, \delta, t \models \bigcirc\varphi$.

Finally, we consider Statement 6. Assume for the sake of contradiction that $\mathrm{M}, \delta, t \models \text{initial} \wedge \text{terminal}$. Then, $\delta[t] = \bar{\mathrm{w}}$ and $\delta[t] \in \mathrm{T}$. By the path definition, it should be the case that $t = 0$. Due to the loop on terminal states, it follows that $\mathrm{w}_0 = \mathrm{w}_1$, which is a contradiction with the path requirement $\mathrm{w}_{t'} \neq \mathrm{w}_0$, for any $t' \geq 1$. □

Statements 1 and 2 specify an agent performs exactly one action in each state. Furthermore, if she does an action, then it must be legal (Statement 3). As a consequence from the path construction, we have that no state can be followed by the initial one (Statement 4) and any formula that holds in a terminal state also holds in the subsequent state (Statement 5). Finally, if a state is the initial one then it is not a terminal state (Statement 6).

We also have tautologies related to the partial order $\preceq$:

Observation. Let M be an ST-model, $\mathsf{i} \in \mathrm{N}$ be an agent, $z = (z_1, z_2, \ldots, z_n)$ be a list of numerical terms (*i.e.*, $z \in \mathcal{L}_Z^n$) and $f \in \mathcal{F}$ be any function such that $f : \mathrm{I}^n \to \mathrm{I}$, for some $n \in \mathrm{I}_{\succ 0}$,

1. $\mathrm{M} \models z_1 \ \Re \ z_1$, for $\Re \in \{\leq, \geq, =\}$
2. $\mathrm{M} \models z_1 \ \Re \ z_2 \wedge z_2 \ \Re \ z_1 \rightarrow z_1 = z_2$, for $\Re \in \{\leq, \geq, =\}$
3. $\mathrm{M} \models z_1 \ \Re \ z_2 \wedge z_2 \ \Re \ z_3 \rightarrow z_1 \ \Re \ z_3$, for $\Re \in \{\leq, \geq, =\}$
4. $\mathrm{M} \models z_i = z_j \leftrightarrow f(z_i, z_{-i}) = f(z_j, z_{-i})$, for any $1 \leq i \leq n$
5. $\mathrm{M} \models z_1 = z_2 \leftrightarrow z_2 = z_1$
6. $\mathrm{M} \models z_1 \ \Re \ z_2 \rightarrow \neg(z_2 \ \Re \ z_1)$, for $\Re \in \{<, >\}$

Proof. Statements 1-6 follow from the definition of $\preceq$ and the abbreviations for the comparison operators. □

Statements 1, 2 and 3 are a consequence from $\preceq$ being reflexive, antisymmetric and transitive, respectively. Two numerical terms are equal if and only if the value obtained after applying f is equal for both terms (Statement 4). Statement 5 says that the equality operator is symmetric and Statement 6 states that the operators representing the relations *smaller-than* and *greater-than* are asymmetric.

Requirements for representing verifiable auctions For verifying a protocol expressed in ADL with respect to mechanism properties, its signature $\mathcal{S} = (\mathrm{N}, \mathrm{G}, \mathcal{B}, \Phi, \mathrm{Y}, \mathrm{I}, \mathcal{F})$ must comply with the following requirements:

- $\mathcal{F}$ should include a function $v_{\mathsf{i}} : \mathcal{B} \times \mathrm{I}^{\mathsf{nm}} \rightarrow \mathrm{I}$ for each agent $\mathsf{i} \in \mathrm{N}$, where $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda})$ denotes the value of β given a joint trade $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$, *i.e.*, $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda})$ represents the value reported for trade $\boldsymbol{\lambda}$ under i's bid β[4];
- There are no duplicate bids in $\mathcal{B}$, that is, there are no two bids $\beta, \beta' \in \mathcal{B}$, such that $\beta \neq \beta'$ and $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda}) = v_{\mathsf{i}}(\beta', \boldsymbol{\lambda})$, for any trade $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$ and any agent i;
- Each payment and trade should be represented as a numerical variable, that is, $\{\mathrm{pay}_{\mathsf{i}}, \mathrm{trade}_{\mathsf{i},j} : \mathsf{i} \in \mathrm{N}, j \in \mathrm{G}\} \subseteq \mathrm{Y}$. The variables $\mathrm{pay}_{\mathsf{i}}$ and $\mathrm{trade}_{\mathsf{i},j}$ denote the value in a state of agent i's payment and her trade for the good j (that is, the number of units of j being bought or sold by i), respectively.

Other functions may as well be included in $\mathcal{F}$. For instance, for indirectly representing market clearing with ADL, one may encode a winner determination function, such that it assigns bids and allocations to trades. Such function is not a compulsory requirement for the bidding language, since we can also directly represent the market clearing through $\mathcal{L}_{\mathsf{ADL}}$-rules[5].

2.3 Verification of ADL-Descriptions

In this section, we explore the general evaluation of auction-based protocols. First, we recall concepts from mechanism design and present their formulation in ADL. As for GDL, we then define well-formulated protocol descriptions.

2.3.1 Direct Mechanisms

A mechanism aggregates agents' preferences and decides for an *outcome* (*e.g.*, an allocation of goods, a result of an election, etc) (Conitzer and Sandholm, 2002). Auctions are a type of mechanism, in which the outcome is described in terms of trades and monetary transfers among the participants. According to a given

[4]For instance, assume the bid β ="buy one apple for 2€ or two for 3€". We can define a bid value function $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda})$ such that the bid $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda}) = 2$ when $\lambda_{\mathsf{i},\mathrm{apple}} = 1$ and $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda}) = 3$ when $\lambda_{\mathsf{i},\mathrm{apple}} = 2$. Other examples are given in the next chapter.

[5]A winner determination function is shown in Section 3.2.1.2 and the protocol presented in Section 3.1 has the market clearing represented through $\mathcal{L}_{\mathsf{ADL}}$-rules.

objective, the goal of Mechanism Design is to design a game (*i.e.*, the mechanism) such that an outcome with desirable features is reached, despite the agents' self interests (Sandholm, 2003). The objective of a mechanism can include, for instance, truthfulness of agents (*i.e.*, strategyproofness), maximization of social welfare (*i.e.*, efficiency), voluntary participation (*i.e.*, individual rationality), and so on.

Let us recall concepts from economics and show how to represent some classical but important objectives (hereafter called *properties*) in ADL, namely budget-balance, efficiency, individual rationality and strategyproofness. As we focus on auctions, we denote the mechanism outcome as a pair $(\boldsymbol{\lambda}, \mathbf{p})$, where $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_j)_{j \in \mathrm{G}}$ is a trade (describing the items being exchanged among the agents) and p_i is the payment for agent i.

The preference of an agent i in N over a trade $\boldsymbol{\lambda}$ is modeled by a preference function $\vartheta_\mathsf{i} : \mathrm{I}^{\mathsf{nm}} \to \mathrm{I}$, where $\vartheta_\mathsf{i} \in V_\mathsf{i}$ and V_i is a finite set of possible preference functions for i. We call V_i the preference space of i. We classically assume that the utility of agent i over an outcome $(\boldsymbol{\lambda}, \mathbf{p})$ is quasi-linear (*i.e.*, the utility's dependence on the payment is separable and linear)[6], defined as $\vartheta_\mathsf{i}(\boldsymbol{\lambda}) - \mathrm{p}_\mathsf{i}$.

We use the value reported in a bid for a given trade to assess whether the bid represents the agent's preference function.

Definition 2.6. Let $\mathcal{B}$ be an action set and V_i be the preference space of agent $\mathsf{i} \in \mathrm{N}$. A bid $\beta \in \mathcal{B}$ *represents* a preference function $\vartheta_\mathsf{i} \in V_\mathsf{i}$, denoted by $\beta \sim_\mathsf{i} \vartheta_\mathsf{i}$, iff $v_\mathsf{i}(\beta, \boldsymbol{\lambda}) = \vartheta_\mathsf{i}(\boldsymbol{\lambda})$, for all trade $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$.

Similarly, an action set may represent a preference space. In this case, exactly one bid in the action set should represent a preference function.

Definition 2.7. Given a set of actions $\mathcal{B}$ and a preference space V_i of agent $\mathsf{i} \in \mathrm{N}$, we say $\mathcal{B}$ represents V_i, denoted by $\mathcal{B} \approx_\mathsf{i} V_\mathsf{i}$ iff for each $\vartheta_\mathsf{i} \in V_\mathsf{i}$ there exists a unique bid $\beta \in \mathcal{B}$ such that $\beta \sim_\mathsf{i} \vartheta_\mathsf{i}$ and for each bid $\beta \in \mathcal{B}$ there exists a preference function $\vartheta_\mathsf{i} \in V_\mathsf{i}$ such that $\beta \sim_\mathsf{i} \vartheta_\mathsf{i}$.

If $\mathcal{B} \approx_\mathsf{i} V_\mathsf{i}$, for each $\vartheta_\mathsf{i} \in V_\mathsf{i}$ we let $\beta_{\vartheta_\mathsf{i}}$ denote the bid $\beta \in \mathcal{B}$ such that $\beta \sim_\mathsf{i} \vartheta_\mathsf{i}$, that is, $\beta_{\vartheta_\mathsf{i}}$ is the bid that represents ϑ_i. Given a preference profile $\boldsymbol{\vartheta}$, let $\boldsymbol{\beta}_{\boldsymbol{\vartheta}} = (\beta_{\vartheta_\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}$ denote the profile of bids representing $\boldsymbol{\vartheta}$.

Notice not all elements of I are feasible values for trades, for instance in a traditional English auction, the trade for each agent should be either 0 or 1 while the interval I could include greater values for encoding the payments. In the following, the possible choices considered for trades in the mechanism are denoted $\Lambda \subseteq \mathrm{I}^{\mathsf{nm}}$.

An indirect mechanism describes the available actions for each agent and an outcome function that maps vectors of actions (also know as *strategies*) into outcomes. In a direct mechanism, each agent's available action consists on reporting preferences from her preference space (Jackson, 2009). Formally, a direct mechanism is defined as follows (Nisan et al., 2007):

[6] Quasilinearity of utilities refers to the fact that the utility function is a linear combination of the preference valuation function and the price paid by the agent. However, the preference function ϑ_i itself can be general.

Definition 2.8. A direct mechanism $(\mathsf{s}, \mathbf{p})$ specifies a social choice function $\mathsf{s} : \prod_{\mathsf{i}\in\mathrm{N}} V_{\mathsf{i}} \to \Lambda$ and a profile of payment functions $\mathbf{p}$, where $\mathsf{p}_{\mathsf{i}} : \prod_{\mathsf{i}\in\mathrm{N}} V_{\mathsf{i}} \to \mathrm{I}$ denotes the amount agent i pays (or receives).

Regardless whether a protocol is multi- or single-stage, we view each step $\mathrm{w}_e \xrightarrow{d^{e+1}} \mathrm{w}_{e+1}$ of a path $\mathrm{w}_0 \xrightarrow{d^1} \mathrm{w}_1 \xrightarrow{d^2} \dots \xrightarrow{d^t} \mathrm{w}_t \xrightarrow{d^{t+1}} \dots$ in an ST-model M as a direct mechanism, where $e \geq 0$ and the social choice and payments are encoded by the update and valuation functions in M. Formally,

Definition 2.9. Given a preference space profile $\boldsymbol{V}$, an ST-model M and a state $\mathrm{w} \in \mathrm{W}$ such that $\mathrm{L}(\mathrm{w}, \mathsf{i}) \approx_{\mathsf{i}} V_{\mathsf{i}}$ for each $\mathsf{i} \in \mathrm{N}$, and let the set of possible trades be defined as $\Lambda = \{(\pi_{\mathrm{Y}}(\mathrm{w}', \mathrm{trade}_{\mathsf{i},j}))_{\mathsf{i}\in\mathrm{N}, j\in\mathrm{G}} : \mathrm{w}' = \mathrm{U}(\mathrm{w}, \boldsymbol{\beta}_{\boldsymbol{\vartheta}}) \;\&\; \boldsymbol{\vartheta} \in \prod_{\mathsf{i}\in\mathrm{N}} V_{\mathsf{i}}\}$. Then state w is a direct mechanism $(\mathsf{s}, \mathbf{p})$, where for each $\boldsymbol{\vartheta} \in \prod_{\mathsf{i}\in\mathrm{N}} V_{\mathsf{i}}$, the outcome is denoted by the valuation of the numerical variables regarding trades and payments in the state $\mathrm{w}' = \mathrm{U}(\mathrm{w}, \boldsymbol{\beta}_{\boldsymbol{\vartheta}})$. The social choice function is

$$\mathsf{s}(\boldsymbol{\vartheta}) = (\boldsymbol{\lambda}_j)_{j\in\mathrm{G}}$$

where $\lambda_{\mathsf{i},j} = \pi_{\mathrm{Y}}(\mathrm{w}', \mathrm{trade}_{\mathsf{i},j})$ for each agent i and good j. The payment function for agent i is

$$\mathsf{p}_{\mathsf{i}}(\boldsymbol{\vartheta}) = \pi_{\mathrm{Y}}(\mathrm{w}', \mathrm{pay}_{\mathsf{i}})$$

The state $\mathrm{w}' = \mathrm{U}(\mathrm{w}, \boldsymbol{\beta}_{\boldsymbol{\vartheta}})$ in the above definition is called an *outcome state*. Any reachable state in an ST-model is an outcome state, except the initial state. For the next subsections, we fix an ST-Model M and a preference space profile $\boldsymbol{V}$ such that $\mathrm{L}(\mathrm{w}, \mathsf{i}) \approx_{\mathsf{i}} V_{\mathsf{i}}$ for each agent i and state $\mathrm{w} \in \mathrm{W}$.

2.3.1.1 Budget-balanced mechanisms

A mechanism is *strongly budget-balanced* (SBB) if the cumulative payment among the bidders is zero, for every preference they may have (Mishra and Sharma, 2018). A mechanism where there is no monetary deficit, that is, where only the designer can earn revenue, is called *weakly budget-balanced* (WBB) (Mishra and Sharma, 2018). This condition is a relaxation from SBB, where the cumulative payment among the bidders cannot be negative.

Definition 2.10. A direct mechanism $(\mathsf{s}, \mathbf{p})$ is strongly budget-balanced (resp. weakly budget-balanced) if for each $\boldsymbol{\vartheta} \in \prod_{\mathsf{i}\in\mathrm{N}} V_{\mathsf{i}}$,

$$\sum_{\mathsf{i}\in\mathrm{N}} \mathsf{p}_{\mathsf{i}}(\boldsymbol{\vartheta}) = 0 \text{ (resp. } \sum_{\mathsf{i}\in\mathrm{N}} \mathsf{p}_{\mathsf{i}}(\boldsymbol{\vartheta}) \geq 0)$$

We denote the condition of a state being SBB by the following ADL-formula:

$$\mathrm{SBB} =_{\mathrm{def}} \mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(\mathrm{pay}_{\mathsf{i}}) = 0$$

The formula WBB is defined similarly:

$$\mathrm{WBB} =_{\mathrm{def}} \mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(\mathrm{pay}_{\mathsf{i}}) \geq 0$$

Remind we consider that each stage in M represents a direct mechanism. The ST-model M is SBB (resp. WBB) if that is the case for all outcome states of all paths in M, that is, if $\mathrm{M} \models \bigcirc\mathrm{SBB}$ (resp. $\mathrm{M} \models \bigcirc\mathrm{WBB}$).

Similarly, we could represent conditions for the balance of trades. *Balance of supply-demand* requires the cumulative of trades for each good to be exactly zero (Larsen et al., 2013). Mechanisms with *free disposal* allow trades to sell more items than are purchased (Parkes and Ungar, 2001), that is, the cumulative of trades for each good must be at most zero.

2.3.1.2 Strategyproof mechanisms

A mechanism is *strategyproof* (SP), or incentive compatible, if each agent i prefers reporting her real preference ϑ_{i} than reporting any other preference ϑ'_{i}, since ϑ_{i} gives her at least the same utility (Nisan et al., 2007).

Definition 2.11. A direct mechanism $(\mathsf{s}, \mathbf{p})$ is strategyproof if for every agent $\mathsf{i} \in \mathrm{N}$, every preference profile $\boldsymbol{\vartheta}$ and every $\vartheta'_{\mathsf{i}} \in V_{\mathsf{i}}$,

$$\vartheta_{\mathsf{i}}(\mathsf{s}(\boldsymbol{\vartheta})) - \mathsf{p}_{\mathsf{i}}(\boldsymbol{\vartheta}) \geq \vartheta_{\mathsf{i}}(\mathsf{s}(\vartheta'_{\mathsf{i}}, \vartheta_{-\mathsf{i}})) - \mathsf{p}_{\mathsf{i}}(\vartheta'_{\mathsf{i}}, \vartheta_{-\mathsf{i}})$$

Now we reformulate this condition in terms of states from an ST-model. Let V_{i} denote the preference space for each agent i.

Given the ST-model M, a path δ in M and a stage $t \geq 0$ in δ, such that $\mathrm{L}(\delta[t], \mathsf{i}) \approx_{\mathsf{i}} V_{\mathsf{i}}$ for each i. For any preference profile $\boldsymbol{\vartheta}$, let $\delta_{\boldsymbol{\vartheta}}$ denote a path such that $\delta[0,t] = \delta_{\boldsymbol{\vartheta}}[0,t]$ and $\theta(\delta_{\boldsymbol{\vartheta}}, t) = \boldsymbol{\beta}_{\boldsymbol{\vartheta}}$ (*i.e.*, $\mathrm{M}, \delta_{\boldsymbol{\vartheta}}, t \models \mathrm{does}(\boldsymbol{\beta}_{\boldsymbol{\vartheta}})$). In other words, $\delta_{\boldsymbol{\vartheta}}$ is a path with the same prefix as δ, but one in which agents report the preferences $\boldsymbol{\vartheta}$ in $\delta_{\boldsymbol{\vartheta}}[t]$ instead of the actions they perform in $\delta[t]$.

We say that $\delta[t]$ is strategyproof if for every $\mathsf{i} \in \mathrm{N}$, every preference profile $\boldsymbol{\vartheta}$ and every $\vartheta'_{\mathsf{i}} \in V_{\mathsf{i}}$, we have that, for some $x \in \mathrm{I}$,

$$\mathrm{M}, \delta_{\boldsymbol{\vartheta}}, t \models \bigcirc\mathrm{sub}(v_{\mathsf{i}}(\beta_{\vartheta_{\mathsf{i}}}, (\mathrm{trade}_{\boldsymbol{j}})_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) = x$$

and

$$\mathrm{M}, \delta_{(\vartheta'_{\mathsf{i}}, \vartheta_{-\mathsf{i}})}, t \models \bigcirc\mathrm{sub}(v_{\mathsf{i}}(\beta_{\vartheta_{\mathsf{i}}}, (\mathrm{trade}_{\boldsymbol{j}})_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) \leq x$$

M is strategyproof if each stage $t \geq 0$ of each path δ in M is strategyproof.

2.3.1.3 Efficient mechanisms

A mechanism is *efficient* (EF) if the social choice function maximizes the (utilitarian) social welfare (Parkes and Ungar, 2001), *i.e.*, the cumulative preference among the agents.

Definition 2.12. A direct mechanism $(\mathsf{s}, \mathbf{p})$ is efficient if for every preference profile $\boldsymbol{\vartheta}$,

$$\sum_{\mathsf{i} \in \mathrm{N}} \vartheta_{\mathsf{i}}(\mathsf{s}(\boldsymbol{\vartheta})) = \max_{\boldsymbol{\lambda} \in \Lambda} \sum_{\mathsf{i} \in \mathrm{N}} \vartheta_{\mathsf{i}}(\boldsymbol{\lambda})$$

Let us express this condition in terms of an ST-model M. The following formula determines whether the current trade maximizes the social welfare:

$$\mathrm{EF}(\boldsymbol{\beta}) =_{\mathrm{def}} \left(\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_j)_{j\in\mathrm{G}})) = \max_{\boldsymbol{\lambda}\in\Lambda}(\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda})))\right)$$

We say M is EF if, after performing a joint action, the trade in the outcome state maximizes agents' preferences, that is $\mathrm{M} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{EF}(\boldsymbol{\beta})$, for every $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$.

2.3.1.4 Individually rational mechanisms

A mechanism is (ex-post) *individually rational* (IR), if agents always get non-negative utility (Nisan et al., 2007). Given a reported preference, in an IR mechanism, the agent's utility when participating is at least as good as if she did not participate (assuming the utility of non-participation is zero). Individual rationality is also known as voluntary participation since it expresses the idea that agents are not forced to participate in the mechanism (Parkes and Ungar, 2001).

Definition 2.13. A direct mechanism $(\mathsf{s}, \mathbf{p})$ is individually rational if for every agent $\mathsf{i} \in \mathrm{N}$, every $\boldsymbol{\vartheta} \in \prod_{\mathsf{r}\in\mathrm{N}} V_{\mathsf{r}}$,

$$\vartheta_{\mathsf{i}}(\mathsf{s}(\boldsymbol{\vartheta})) - \mathsf{p}(\boldsymbol{\vartheta}) \geq 0$$

We use the following ADL-formula to denote whether a state is IR:

$$\mathrm{IR}(\boldsymbol{\beta}) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N}} \mathrm{sub}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_j)_{j\in\mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) \geq 0$$

The ST-model M is IR if performing a joint action leads to an individually rational state, that is, $\mathrm{M} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{IR}(\boldsymbol{\beta})$, for every $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$.

The properties described in this section are classical in mechanism design, since they describe desirable features of the outcome. The objective of a mechanism may include a combination of different properties. However, well-known impossibility results restrict the feasible combination of such properties: no mechanism can be efficient, strongly budget-balanced and individual-rational (Myerson and Satterthwaite, 1983) and no mechanism can be efficient, incentive compatible and strongly budget-balanced (Green and Laffont, 1979).

In this chapter, we describe how to verify ST-models by considering that each stage is a direct mechanism, that is, an iterative protocol is treated as a sequence of (independent) direct mechanisms. The revelation principle (Nisan et al., 2007) states that any indirect mechanism that implements a function in dominant strategies can be converted into a strategyproof direct mechanism. For this reason, considering direct mechanisms is of first interest.

If we want to verify an ST-model M as an (unique) indirect mechanism, we need to evaluate properties considering the final outcome, that is, on the terminal

states of each path. The classical approach in mechanism design requires properties to hold in strategic equilibrium rather than for all possible outcomes. As ADL does not involve quantification over strategies, we need meta-reasoning to capture the strategic equilibrium for a given solution concept (such as Nash or dominant strategy equilibrium). The reader may refer to (Nisan et al., 2007) for a discussion on the problem of finding strategic equilibria.

In the next subsection, we will focus on properties that ensure well-formed descriptions in ADL. This properties are not related to the mechanism outcome but require protocols to be playable and eventually end.

2.3.2 Well-Formed Protocols

Love et al. (2006) introduced constraints for games used in GGP. This constraints constitute desirable features of games described in GDL by ensuring their descriptions to be meaningful (or *well-formed*), in the sense that games are playable, eventually terminate and are weakly winnable by any player.

We rephrase the constraints for termination and playability in terms of ST-models. First, termination refers to whether each path from an ST-model reaches a terminal state.

Definition 2.14. An ST-model M *terminates* if each path δ in M is complete, that is, $\delta[t] \in \mathrm{T}$, for some $t > 0$.

Playability means there is a legal action for each agent to take in each moment of the auction.

Definition 2.15. An ST-model M is *playable* if $\mathrm{L}(\mathrm{w}, \mathsf{i}) \neq \emptyset$ for each *reachable* state $\mathrm{w} \in \mathrm{W}$ and agent $\mathsf{i} \in \mathrm{N}$.

Weak winnability means that, for each agent, there is a sequence of joint actions that leads to a terminal state where the goal value is maximal. In the logical formulations of GDL, weak winnability means that every player has a chance to win (Zhang, 2018; Ruan et al., 2009), that is, there exists a path to a winning state. In ADL, weak winnability follows from our assumption that each stage represents a direct mechanism. Since we understand the agent's preferences as represented by legal actions, there exists a joint action in each stage leading to a state that maximizes her utility among the possible outcomes. Hence, we do not focus on weak winnability for determining whether a protocol is well-formed.

A well-formed protocol is a set of rules in ADL whose model is an ST-model that satisfies both termination and playability.

Definition 2.16. Given an ST-Model M and a finite set of ADL-formulae $\Sigma \subset \mathcal{L}_{\mathsf{ADL}}$, Σ is a *well-formed* protocol over M if M is a model of Σ, M terminates and it is playable.

It is possible to have different descriptions that are well-formed with respect to the same model (*e.g.*, due to redundancy). In fact, a given auction description may

also be sound in respect to several models. Investigating minimal descriptions is an interesting non trivial open question. A potential path is to characterize the minimum equivalent of an original ST-model, that is, the canonical model. This problem was explored for GDL (Jiang et al., 2019), where they use the notion of bisimulation equivalence between ST-models. In a recent paper, Zhang (Zhang, 2020) investigates the equivalence of two GDL-descriptions when they describe games behaviorally the same.

The automated verification of strategyproofness and termination requires meta-reasoning over the possible paths of a ST-Model. However, for a number properties (such as efficiency and individual rationality), the problem of analyzing a stage as a direct mechanism boils down to model checking ADL-formulae.

2.4 Model Checking

Now we examine the complexity of the problem of deciding whether an ADL formula is true with respect to a model, a path, and a stage in the path.

Definition 2.17. The *model-checking problem* for ADL is the following: Given an ST-Model M, a path δ in M, a stage $t \geq 0$ in δ and a formula $\varphi \in$ ADL, determine whether $\mathrm{M}, \delta, t \models \varphi$ or not.

In the next sections, we show that the model-checking problem for ADL is decidable in polynomial-time deterministic Turing machine (PTIME) when functions in $\mathcal{F}$ can be computed in polynomial time.

2.4.1 Upper Bound

Let $\varphi \in$ be a formula and M be an ST-Model over $\mathcal{S}$. We say that ψ is a subformula of φ if either (i) $\psi = \varphi$; (ii) φ is of the form $\neg\varphi$, or $\bigcirc\psi$ and ψ is a subformula of φ; or (iii) φ is of the form $\varphi \wedge \varphi'$ and ψ is a subformula of either φ or φ'. We denote *Subformula*(φ) as the set of all subformulae of φ.

Theorem 2.1. *Assuming that functions in $\mathcal{F}$ can be computed in polynomial time, the model-checking problem for* ADL *is in* PTIME.

Proof. Assume the functions in $\mathcal{F}$ can be computed in polynomial time. Algorithm *modelCheck* works in the following way: first it gets all subformulae of φ and orders them in a vector S by ascending length. Thus, $S(|\varphi|) = \varphi$, i.e., the position $|\varphi|$ in S corresponds to the formula φ itself, and if φ_i is a subformula of φ_j then $i < j$. An induction on S labels each subformula φ_i depending on whether or not φ_i is true in $\delta[j]$ under M. Since functions in $\mathcal{F}$ can be computed in polynomial time, if φ_i does not have any subformula, its truth value is obtained directly from the model. Since S is ordered by formulae length, if φ_i is either of the form $\varphi' \wedge \varphi''$ or $\neg\varphi'$ the algorithm labels φ_i according to the label assigned to φ' and/or φ''. If φ_i is of the form $\bigcirc\varphi'$ then its label is recursively defined according to φ' truth value in the stage $t + 1$. As Algorithm *modelCheck* visits each subformula at most once,

Algorithm 1 *modelCheck*$(\mathrm{M}, \delta, t, \varphi)$

Input: an ST-model $\mathrm{M} = (\mathrm{W}, \bar{\mathrm{w}}, \mathrm{T}, \mathrm{L}, \mathrm{U}, \pi_\Phi, \pi_\mathrm{Y})$, a path δ in M, a stage $t \geq 1$ in δ and a formula $\varphi \in \mathcal{L}_{\mathsf{ADL}}$.

Output: **true** if $\mathrm{M}, \delta, t \models \varphi$, and **false** otherwise

$S \leftarrow$ *Subformula*(φ) ordered by ascending length
Let $isTrue[1, \cdots, |\varphi|]$ be a boolean array initiated with *true* values
for $i \leftarrow 1$ to $|\varphi|$ **do**
    $\varphi \leftarrow S[i]$
    switch the formula type of φ **do**
        case φ is of the form $\varphi' \wedge \varphi''$
            $isTrue[i] \leftarrow isTrue[getIndex(S, \varphi')] \wedge isTrue[getIndex(S, \varphi'')]$
        case φ is of the form $\neg\varphi'$
            $isTrue[i] \leftarrow \neg isTrue[getIndex(S, \varphi')]$
        case φ is of the form $\bigcirc\varphi'$
            $isTrue[i] \leftarrow isTrue[i] \wedge \mathrm{modelCheck}(\mathrm{M}, \delta, j+1, \varphi')$
        case φ is atomic
            $isTrue[i] \leftarrow \mathrm{M}, \delta, t \models \varphi$
return $isTrue[|\varphi|]$

and the number of subformulas is not greater than the size of φ, the algorithm can clearly be implemented in a polynomial-time deterministic Turing machine with PTIME. □

2.4.2 Lower Bound

Now let us characterize a lower bound of the complexity of model-checking ADL. To do so, we reduce the model-checking problem for GDL, which is known to be PTIME (consequence of Lemma 2 by Jiang et al. (2021)). This reduction is computable in logarithmic space in the size of the input.

Theorem 2.2. *The model-checking problem for* ADL *is* PTIME-HARD.

Proof. First, we recall GDL definitions. By convenience, we refer to the GDL formalization presented in (Jiang et al., 2019). Let $\mathcal{S}_{\mathrm{GDL}} = (\mathrm{N}, \mathcal{B}, \Phi)$ be a game signature, where Φ is a propositional set, N is a set of agents, and $\mathcal{B}$ is the action set. Let $\mathrm{M}_{\mathrm{GDL}} = (\mathrm{W}, \bar{\mathrm{w}}, \mathrm{T}, \mathrm{L}, \mathrm{U}, g, \pi_\Phi)$ be a GDL model over $\mathcal{S}_{\mathrm{GDL}}$, where W, $\bar{\mathrm{w}}$, T, L and π_Φ are defined in the same way as in a ADL model (see Definition 2.2), $\mathrm{U} : \mathrm{W} \times \mathcal{B}^{|\mathrm{N}|} \to \mathrm{W} \setminus \{\bar{\mathrm{w}}\}$ is the update function and $g : \mathrm{N} \to 2^{\mathrm{W}}$ is the goal function.

A path δ in $\mathrm{M}_{\mathrm{GDL}}$ is an infinite sequence of states and joint actions, defined in the same way as Definition 2.3. Let δ be a path in $\mathrm{M}_{\mathrm{GDL}}$, a formula φ in $\mathcal{L}_{\mathrm{GDL}}$ is defined by the following BNF:

$$\varphi ::= p \mid \mathrm{initial} \mid \mathrm{terminal} \mid \mathrm{legal}_{\mathsf{r}}(a) \mid \mathrm{wins}(\mathsf{r}) \mid \mathrm{does}_{\mathsf{r}}(a) \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi$$

where $p \in \Phi$, $\mathsf{r} \in \mathrm{N}$ and $a \in \mathcal{B}$.

For any stage $t \geq 0$ and formula $\varphi \in \mathcal{L}_{\mathrm{GDL}}$, the semantics of $\mathcal{L}_{\mathrm{GDL}}$ are similar to Definition 2.5, except by the following case: $\mathrm{M}_{\mathrm{GDL}}, \delta, t \models \mathrm{wins}(\mathsf{r})$ iff $\delta[t] \in g(\mathsf{r})$.

Given a stage $t > 0$ and a formula $\varphi \in \mathcal{L}_{\mathrm{GDL}}$, we show how to construct an ADL model $\mathrm{M}_{\mathsf{ADL}}$, such that $\mathrm{M}, \delta, t \models \varphi$ iff $\mathrm{M}_{\mathrm{GDL}}, \delta, t \models \varphi$.

Let $\mathcal{S}_{\mathsf{ADL}} = (\mathrm{N}, \emptyset, \mathcal{A}, \Phi', \emptyset, [0,0], \emptyset)$ be the auction signature, where $\Phi' = \Phi \cup \{\mathrm{wins}(\mathsf{r}) : \mathsf{r} \in \mathrm{N}\}$. We define the ADL model $\mathrm{M}_{\mathsf{ADL}} = (\mathrm{W}, \bar{\mathrm{w}}, \mathrm{T}, \mathrm{L}, \mathrm{U}, \pi'_{\Phi}, \emptyset)$, where $\mathrm{W}, \bar{\mathrm{w}}, \mathrm{T}$ and U are the same as in $\mathrm{M}_{\mathrm{GDL}}$. The valuation function for state propositions is defined as follows: $\pi'_{\Phi}(\mathrm{w}) = \pi_{\Phi}(\mathrm{w}) \cup \{\mathrm{wins}(\mathsf{r}) : \mathrm{w} \in g(\mathsf{r}) \ \& \ \mathsf{r} \in \mathrm{N}\}$, for each $\mathrm{w} \in \mathrm{W}$.

Since $\mathrm{U}(\mathrm{w}, \boldsymbol{d}) \in \mathrm{W}$, for each $\mathrm{w} \in \mathrm{W}$ and $\boldsymbol{d} \in \mathcal{B}^{|\mathrm{N}|}$, we have that U is an update function in accordance with Definition 2.2. Furthermore, if δ is a path in $\mathrm{M}_{\mathrm{GDL}}$ then it is a path in $\mathrm{M}_{\mathsf{ADL}}$. Since $\{\mathrm{wins}(\mathsf{r}) : \mathsf{r} \in \mathrm{N}\} \subseteq \Phi'$, we have that $\mathcal{L}_{\mathrm{GDL}} \subseteq \mathcal{L}_{\mathsf{ADL}}$.

Given $\varphi \in \mathcal{L}_{\mathrm{GDL}}$, if φ is not in the form $\mathrm{wins}(\mathsf{r})$, then it is straightforward that $\mathrm{M}_{\mathrm{GDL}}, \delta, t \models \varphi$ iff $\mathrm{M}_{\mathsf{ADL}}, \delta, t \models \varphi$. Now we consider the case were $\varphi \in \mathcal{L}_{\mathrm{GDL}}$ is in the form $\mathrm{wins}(\mathsf{r})$, for some $\mathsf{r} \in \mathrm{N}$. Assume $\mathrm{M}_{\mathrm{GDL}}, \delta, t \models \mathrm{wins}(\mathsf{r})$ iff $\delta[t] \in g(\mathsf{r})$ iff $\mathrm{wins}(\mathsf{r}) \in \Phi'(\delta[t])$ iff $\mathrm{M}_{\mathsf{ADL}}, \delta, t \models \mathrm{wins}(\mathsf{r})$.

□

Theorem 2.2 shows that Algorithm *modelCheck* is optimal when the functions in $\mathcal{F}$ can be computed in polynomial time. As for GDL, the ST-model may have an exponential size with respect to the number of atomic propositions used (*e.g.*, the number of agents and items).

2.5 Conclusion

ADL is useful for representing the rules of an auction as well as for verifying a number of properties. For discussing its generality, we refer again to the six auction dimensions from Kalagnanam and Parkes (2004)'s classification:

1. Resources: ADL can represent auction variants with single and multiple units as well as multiple types of goods;

2. Market structure: ADL represents single and double-sided auctions in addition to exchange protocols. However, we cannot represent dynamic sets of agents and goods;

3. Preference structure: as we are concerned with the general representation of auctions, we focus on how agents can *express* their preferences rather than on the underlying structures of their utility functions. Following the literature on Mechanism Design (Parkes and Ungar, 2001), we consider agents with quasi-linear utilities. In such a case, the agents' utilities are based on a preference function over the outcomes and her payment. For a discussion on the hierarchy of preference functions, the reader may refer to Feige et al. (2015);

4. Bid structure: ADL can be used alongside different bidding languages. We define requirements for the bidding set. In the next chapter, we illustrated how to employ different bidding structures in ADL;

5. Market clearing: ADL can encode single and multi-sourcing auctions;

6. Information feedback: in ADL, direct mechanisms are described by a one-stage protocol and indirect mechanisms are represented by iterative protocols.

Hence, ADL addresses all these dimensions and is general enough to represent most settings. Notice that, similar to GDL, ADL focuses on deterministic and perfect information protocols. GDL-II is an extension for handling imperfect information (Thielscher, 2011) and may be considered for auctions such as iterative sealed-bid auctions. These are however less common.

Different from the related work (Rolli et al., 2006), our proposal is flexible and not restricted to a specific bidding language, as one may use different bidding languages with ADL for defining distinct protocols.

Beyond the auction setting, ADL is also able to represent several kinds of resource allocation problems: as noticed by Chevaleyre et al. (2006), auctions can be seen as a subdivision of allocation mechanisms. The main characteristics of an auction are (i) central authority (the auctioneer), (ii) monetary transfer among participants, and (iii) agents' preferences expressed through bids. All these key features are expressible in ADL.

Similar to ADL, the semantics of GDL is based on fixed paths. Thus, constraints for well-formed descriptions cannot be encoded through formulae using the standard formulation of GDL. Zhang (2018) proposes a GDL-based modal logic to enable reasoning over game descriptions, with which one can express conditions such as playability, termination and winnability. However, the work does not investigate the complexity of verifying formulae in this modal logic. Ruan et al. (2009) use ATL (Alur et al., 2002) to reason about GDL-specified games. They prove that the problem of interpreting ATL formulae over propositional GDL descriptions is EXPTIME-complete and show how to use ATL for the verification of well-formedness conditions, which might or might not hold on various games. Deciding whether a GDL description is well-formed is undecidable in general, since deciding whether a description leads to games that always terminates would solve the halting problem for a Turing machine (Saffidine, 2014).

As shown in Section 2.3, we can encode different properties of direct mechanisms as ADL-formulae (*e.g.*, individual rationality and budget balance). Although we cannot represent strategyproofness and the constraints for well-formed descriptions entirely as ADL-formulae, we are still able to infer them by meta-reasoning over the model specification.

Other logical languages are suitable for encoding properties. Termination and playability, for instance, can be written as ATL-formulae (Alur et al., 2002). As SL (Chatterjee et al., 2010) includes quantification over strategies, it allows the evaluation of games in strategic equilibria. Therefore, it can be used to encode properties

for indirect mechanisms as logical formulae, as we shall see in Chapter 5. However, the expressivity of such languages brings a computational cost. The model-checking problem for ATL^* with perfect information is in PSPACE in the memoryless case and deterministic double exponential time with perfect recall (Schobbens, 2004). As for SL, the model-checking is NONELEMENTARY with respect to the size of the specification. More specifically, it is k-EXPSPACE-hard in the alternation number k of quantifications in the specification (Mogavero et al., 2014). The model-checking problem for ADL is in PTIME in the size of the formula. For this reason, we believe ADL provides a reasonable cost-benefit for expressing and evaluating general auctions. As a drawback for ATL, SL and ADL, representing auctions as concurrent game structures or state-transition models may require exponential size.

In the next chapter, we illustrate the use of ADL for specifying different, although representative, types of auctions: a simultaneous ascending auction, a Vickrey–Clarke–Groves mechanism and an iterative combinatorial exchange.

Chapter 3

Representative Auctions in ADL

In the previous chapter, we proposed ADL a logical language for the specification of auctions. We also characterized the well-formedness of auction descriptions and showed how to evaluate properties of direct mechanisms using ADL. Now, we illustrate the generality of ADL by modeling different auction-based markets. We focus on two auction types: simultaneous ascending auction and combinatorial exchange as we believe they are representative to demonstrate the expressive power of ADL. The first generalizes English auctions to multiple items and uses a simple bidding language for stating the price an agent is willing to pay for each item. The second type, combinatorial exchange, is a generalization of combinatorial and double-sided auctions. In this case, we show how to use a tree-based bidding language for elicitation of preferences over logical combinations of goods in two-sided markets. We consider an one-shot and an iterative variants of this auction type. Finally, each of those protocols is evaluated in terms of their well-formedness and the properties from direct mechanisms described in Section 2.3.

3.1 Simultaneous Ascending Auction

Let us now consider the simultaneous ascending auction (SAA), which is a single-side and single-unit protocol similar to the traditional English auction, except that several goods are sold at the same time, and that the participants simultaneously bid for any number of goods they want. According to Cramton (2011):

> "The simultaneous ascending auction (and its variants) will remain the best method for auctioning many related items in a wide range of circumstances, even settings where some of the goods are complements for some bidders."

To represent the SAA with n types of goods and m agents, we first describe the auction signature, written $\mathcal{S}_{sa} = (\mathrm{N}, \mathrm{G}, \mathcal{B}, \Phi, \mathrm{Y}, \mathrm{I}, \mathcal{F})$, where $\mathrm{N} = \{1, \ldots, \mathsf{m}\}$, $\mathrm{G} = \{1, \ldots, \mathsf{n}\}$, $\Phi = \{\mathrm{sold}_j, \mathrm{bid}_{\mathsf{i},j} : j \in \mathrm{G} \;\&\; \mathsf{i} \in \mathrm{N}\}$, $\mathrm{Y} = \{\mathrm{price}, \mathrm{price}_j, \mathrm{trade}_{\mathsf{i},j}, \mathrm{pay}_i : j \in \mathrm{G} \;\&\; \mathsf{i} \in \mathrm{N}\}$ and $\mathrm{I} \subset \mathbb{N}$. The propositions sold_j and $\mathrm{bid}_{\mathsf{i},j}$ represent whether the good j was sold and whether i is bidding for j, resp. The variables price and price_j specify the current price for any unsold good and the selling price for j, resp. Agents may specify the value they are willing to pay for each good in a given state. The action set is defined as follows: $\mathcal{B} = \{(p_1, \ldots, p_{\mathsf{m}}) : p_j \in \mathrm{I}_{\geq 0}, 1 \leq j \leq \mathsf{m}\}$, where p_j denotes the price for good j. $\mathcal{F}$ includes the functions previously introduced

(*e.g.*, $\text{sum}(z_1, z_2)$, $\max(z_1, z_2)$). It also contains $v_{\mathsf{i}} : \mathcal{B} \times \mathrm{I}^{\mathsf{nm}} \to \mathrm{I}$, for each agent $\mathsf{i} \in \mathrm{N}$. This function is defined as follows:

$$v_{\mathsf{i}}((p_1, \ldots, p_{\mathsf{m}}), \boldsymbol{\lambda}) = \sum_{j \in \mathrm{G}} \lambda_{\mathsf{i},j} \cdot p_j$$

for a trade $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$ and a bid $(p_1, \ldots, p_{\mathsf{m}}) \in \mathcal{B}$.

Each instance of a SAA is specific and defined with respect to $\mathcal{B}$, I and the constant values $\mathsf{inc}, \mathsf{m}, \mathsf{n} \in \mathrm{I}_{\succ 0}$ and $\mathsf{start} \in \mathrm{I}_{\succeq 0}$, representing the number of agents and the quantity of distinct good types, the bid increment, and the starting price, respectively. Then, the rules of an SAA are formulated by ADL-formulae as shown in Figure 3.1.

1. $\text{initial} \leftrightarrow \text{price} = \mathsf{start} \wedge \bigwedge_{j \in G} (\text{price}_j = \mathsf{start} \wedge \bigwedge_{\mathsf{i} \in \mathrm{N}} (\neg \text{bid}_{\mathsf{i},j} \wedge \text{trade}_{\mathsf{i},j} = 0))$
2. $\text{sold}_j \leftrightarrow \bigvee_{\mathsf{i} \in \mathrm{N}} \text{trade}_{\mathsf{i},j} = 1$, for each $j \in \mathrm{G}$
3. $\text{terminal} \leftrightarrow \neg\text{initial} \wedge \bigwedge_{j \in \mathrm{G}} (\text{sold}_j \vee \bigwedge_{\mathsf{i} \in \mathrm{N}} \neg \text{bid}_{\mathsf{i},j})$
4. $\bigcirc(\text{trade}_{\mathsf{i},j} = 1 \leftrightarrow \text{bid}_{\mathsf{i},j} \wedge \bigwedge_{\mathsf{r} \in \mathrm{N} \setminus \{i\}} \neg \text{bid}_{r,j})$, for each $\mathsf{i} \in \mathrm{N}$, $j \in \mathrm{G}$
5. $\bigcirc(\text{trade}_{\mathsf{i},j} = 0 \leftrightarrow \neg(\text{bid}_{\mathsf{i},j} \wedge \bigwedge_{\mathsf{r} \in \mathrm{N} \setminus \{i\}} \neg \text{bid}_{r,j}))$, for each $\mathsf{i} \in \mathrm{N}$, $j \in \mathrm{G}$
6. $\text{legal}_{\mathsf{i}}(p_1, \ldots, p_{\mathsf{m}}) \leftrightarrow \bigwedge_{j \in \mathrm{G}} ((p_j = 0 \wedge \text{trade}_{\mathsf{i},j} = 0) \vee (p_j = \text{sum}(\text{price}, \mathsf{inc}) \wedge \neg \text{sold}_j) \vee (p_j = \text{price}_j \wedge \text{trade}_{\mathsf{i},j} = 1))$, for each $\mathsf{i} \in \mathrm{N}$, $p_1, \ldots, p_{\mathsf{m}} \in \{x : 0 \leq x < z_{\max} - \mathsf{inc}\}$
7. $\neg\text{terminal} \wedge \text{price} = x \to \bigcirc\text{price} = \text{sum}(x, \mathsf{inc})$, for each $x \in \mathrm{I}_{\succeq 0}$
8. $\neg\text{terminal} \wedge \text{price}_j = x \to \bigcirc((\text{price}_j = x \wedge \text{sold}_j) \vee (\text{price}_j = \text{sum}(x, \mathsf{inc}) \wedge \neg \text{sold}_j))$, for each $j \in \mathrm{G}$, $x \in \mathrm{I}_{\succeq 0}$
9. $\bigcirc\text{bid}_{\mathsf{i},j} \leftrightarrow (\text{does}_{\mathsf{i}}(p_1, \ldots, p_{\mathsf{m}}) \wedge p_j \neq 0) \vee (\text{bid}_{\mathsf{i},j} \wedge \text{terminal})$, for each $\mathsf{i} \in \mathrm{N}$, $j \in \mathrm{G}$ and some $p_1, \ldots, p_{\mathsf{m}} \in \mathrm{I}_{\succeq 0}$
10. $\text{pay}_{\mathsf{i}} = \text{sum}_{j \in \mathrm{G}}(\text{times}(\text{price}_j, \text{trade}_{i,j}))$, for each $\mathsf{i} \in \mathrm{N}$

Figure 3.1: Simultaneous Ascending Auction represented by Σ_{sa}

In the initial state, no agent is bidding, no trade is performed and the prices have the value start (Rule 1). A good is sold if it is traded to some agent (Rule 2). In a terminal state, all the goods are either sold or no one is bidding for them (Rule 3). A good will be traded to an agent in the next state if she is currently the only active bidder for this item, otherwise there is no trade (Rules 4-5). For each good, an agent can either bid the value 0, an increment on the current price (for unsold goods) or repeat her winning bid for this good (Rule 6). In a non-terminal state, the

propositions and numerical variables are updated as follows: (i) the current price increases, (ii) the selling price increases for unsold goods, and (iii) the active bidders for each good are updated with respect to their bids (Rules 7-9). The payment for an agent is the cumulative value of the selling price for her traded goods (Rule 10). Let Σ_{sa} be the set of Rules 1-10.

3.1.1 Representing as a model

Next, we address the model representation of the SAA. Let $\mathcal{M}_{sa}$ be the set of ST-models M_{sa} defined for any constant values $\mathsf{start} \in \mathrm{I}_{\succeq 0}$ and $\mathsf{inc}, \mathsf{n}, \mathsf{m} \in \mathrm{I}_{\succ 0}$. Each M_{sa} is defined as follows:

- $\mathrm{W} = \{\langle (\boldsymbol{b_j})_{j\in \mathrm{G}}, (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}, p, (p_j)_{j\in \mathrm{G}}\rangle : b_{\mathsf{i},j}, \lambda_{\mathsf{i},j} \in \{0,1\}$ & $p, p_j \in \mathrm{I}_{\succeq 0}$ & $\mathsf{i} \in \mathrm{N}$ & $j \in \mathrm{G}\}$, where $b_{\mathsf{i},j}$ denotes whether agent i is bidding for good j, $\lambda_{\mathsf{i},j}$ specifies the number of goods with type j traded for agent i, p denotes the current price and p_j represents the selling price for j;

- $\bar{\mathrm{w}} = \langle 0, \ldots, 0, 0, \ldots, 0, \mathsf{start}, \mathsf{start}, \ldots, \mathsf{start}\rangle$, in the initial state, there is no trade or active agent and the prices are start;

- $\mathrm{T} = \{\mathrm{w} : \mathrm{w} = \langle (\boldsymbol{b_j})_{j\in \mathrm{G}}, (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}, p, (p_j)_{j\in \mathrm{G}}\rangle \in \mathrm{W} \setminus \{\bar{\mathrm{w}}\}$ & for all $j \in \mathrm{G}$, either (i) $\lambda_{\mathsf{i},j} = 1$ for some $\mathsf{i} \in \mathrm{N}$ or (ii) $\mathrm{bid}_{\mathsf{i},j} = 0$, for all $\mathsf{i} \in \mathrm{N}\}$, the terminal states are the ones where every good was sold or there is no bidder interested on purchasing it;

- $\mathrm{L} = \{(\mathrm{w}, \mathsf{i}, (p_j)_{j\in \mathrm{G}}) : \mathsf{i} \in \mathrm{N}$ & $\mathrm{w} = \langle (\boldsymbol{b_j})_{j\in \mathrm{G}}, (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}, p, (p_j)_{j\in \mathrm{G}}\rangle \in \mathrm{W}$ & for all $j \in \mathrm{G}$, and all $0 \leq p_j < z_{\max} - \mathsf{inc}$ such that either (i) $p_j = 0$ & $\lambda_{\mathsf{i},j} = 0$ or (ii) $p_j = p + \mathsf{inc}$ & $\lambda_{\mathsf{r},j} \neq 1$, for all $\mathsf{r} \in \mathrm{N}$ or (iii) $p_j = p_j$ & $\lambda_{\mathsf{i},j} = 1\}$, that is, agents can choose to raise their bid or to give up of unsold goods, if an agent bought a good, she must keep her bid for it;

- For every $\mathrm{w} = \langle (\boldsymbol{b_j})_{j\in \mathrm{G}}, (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}, p, (p_j)_{j\in \mathrm{G}}\rangle$ in W and all $\boldsymbol{d} \in \mathcal{B}^{\mathsf{m}}$, U is defined as follows: if $\mathrm{w} \in \mathrm{T}$ then $\mathrm{U}(\mathrm{w}, \boldsymbol{d}) = \mathrm{w}$. Otherwise, $\mathrm{U}(\mathrm{w}, \boldsymbol{d}) = \langle (\boldsymbol{b'_j})_{j\in \mathrm{G}}, (\boldsymbol{\lambda'_j})_{j\in \mathrm{G}}, p', (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}\rangle$, where for every $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$ each component is updated as follows,

  (i). $b'_{\mathsf{i},j} = 1$ iff $d_i \neq 0$; and $b'_{\mathsf{i},j} = 0$ otherwise;

  (ii). $\lambda'_{\mathsf{i},j} = 1$ iff $b'_{\mathsf{i},j} = 1$ and for all $\mathsf{r} \in \mathrm{N} \setminus \{i\}, b'_{\mathsf{i},j} \neq 1$; and $\lambda'_{\mathsf{i},j} = 0$ otherwise;

  (iii). $p' = p + \mathsf{inc}$;

  (iv). $p'_j = p_j + \mathsf{inc}$ iff $\lambda'_{r,j} = 0$ for all $\mathsf{r} \in \mathrm{N}$; and $p'_j = p_j$ otherwise.

- For each $\mathrm{w} = \langle (\boldsymbol{b_j})_{j\in \mathrm{G}}, (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}, p, (p_j)_{j\in \mathrm{G}}\rangle$ in W, $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$,

  (i). $\pi_{\mathrm{Y}}(\mathrm{w}, \mathrm{trade}_{\mathsf{i},j}) = \lambda_{\mathsf{i},j}$;

  (ii). $\pi_{\mathrm{Y}}(\mathrm{w}, \mathrm{price}) = p$;

(iii). $\pi_{\mathrm{Y}}(\mathrm{w}, \mathrm{price}_j) = p_j$;

(iv). $\pi_{\mathrm{Y}}(\mathrm{w}, \mathrm{pay}_{\mathsf{i}}) = \sum_{j \in \mathrm{G}} p_j \cdot \lambda_{\mathsf{i},j}$

- For each $\mathrm{w} \in \mathrm{W}$, $\pi_{\Phi}(\mathrm{w}) = \{\mathrm{sold}_j : \lambda_{\mathsf{i},j} = 1 \ \& \ j \in \mathrm{G} \ \& \ \mathsf{i} \in \mathrm{N}\} \cup \{\mathrm{bid}_{\mathsf{i},j} : b_{\mathsf{i},j} = 1 \ \& \ j \in \mathrm{G} \ \& \ \mathsf{i} \in \mathrm{N}\}$.

Hereafter, we assume an instance of $\mathrm{M}_{sa} \in \mathcal{M}_{sa}$ and Σ_{sa} for some $\mathsf{start} \in \mathrm{I}_{\succeq 0}$ and $\mathsf{inc}, \mathsf{n}, \mathsf{m} \in \mathrm{I}_{\succ 0}$.

Example 3.1. Let $\mathrm{M}_{sa} \in \mathcal{M}_{sa}$, such that $\mathsf{start} = 1$ and $\mathsf{inc} = 1$. We assume there are two agents in N, denoted by i and s, and two types of goods in G, denoted by a and b. Figure 3.2 illustrates a path in M_{sa}, showing the value of the numerical variables and the propositions that hold in each state. For convenience, we omit the numerical variables when their value is 0. In state w_0, agents r and s bid for good a, but only agent r bids for good b. In state w_1, since r is the only bidder for b, it is sold to her. Agent r cannot change her bid for b and s can no longer bid for it. In w_1, only agent s accepts to increase her bid for a. In state w_2, a is sold to s. Since all the goods were sold, this state is terminal.

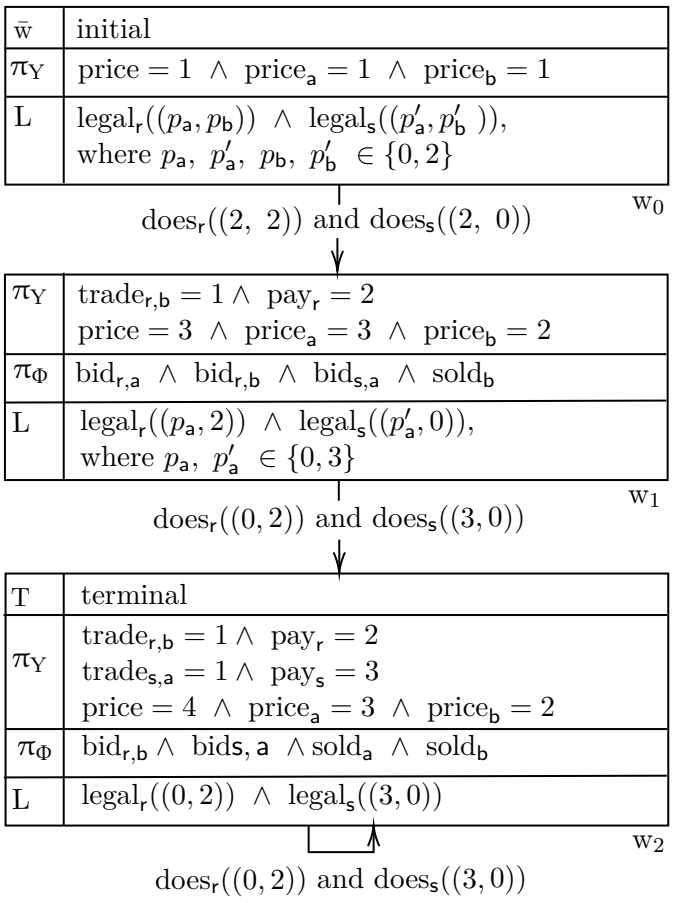

Figure 3.2: A Path in M_{sa}, with 2 bidders and 2 goods

3.1.2 Evaluating the protocol

Let us now evaluate the protocol. First, we show that Σ_{sa} is a sound representation of M_{sa}.

Lemma 3.1. M_{sa} *is an ST-model and it is a model of* Σ_{sa}.

Proof. (Sketch) It is routine to check that M_{sa} is actually an ST-model. Given a path δ, any stage t of δ in M_{sa}, we need to show that $\mathrm{M}_{sa}, \delta, t \models \varphi$, for each $\varphi \in \Sigma_{sa}$. Let us verify Rule 1. Assume $\mathrm{M}_{sa}, \delta, t \models \text{initial}$ iff $\delta[t] = \bar{\mathrm{w}}$. By the definition of $\bar{\mathrm{w}}$, π_Φ and π_Y, we have $\pi_\mathrm{Y}(\bar{\mathrm{w}}, \mathrm{price}) = \mathsf{start}$, $\pi_\mathrm{Y}(\bar{\mathrm{w}}, \mathrm{price}_j) = \mathsf{start}$, $\mathrm{bid}_{\mathsf{i},j} \notin \pi_\Phi(\bar{\mathrm{w}})$ and $\mathrm{trade}_{\mathsf{i},j} = 0$, for all $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$. Thus, $\mathrm{M}_{sa}, \delta, t \models \text{initial}$ iff $\mathrm{M}_{sa}, \delta, t \models \mathrm{price} = \mathsf{start} \wedge \bigwedge_{j\in\mathrm{G}}(\mathrm{price}_j = \mathsf{start} \wedge \bigwedge_{\mathsf{i}\in\mathrm{N}}(\neg\mathrm{bid}_{\mathsf{i},j} \wedge \mathrm{trade}_{\mathsf{i},j} = 0))$.

Now we verify Rule 2. Let $j \in \mathrm{G}$ be a good type. Assume $\mathrm{M}_{sa}, \delta, t \models \mathrm{sold}_j$ iff $\mathrm{sold}_j \in \pi_\Phi(\delta[t])$ iff $\pi_\mathrm{Y}(\delta[t], \mathrm{trade}_{\mathsf{i},j}) = 1$ for some $j \in \mathrm{G}$ iff $\mathrm{M}_{sa}, \delta, t \models \bigvee_{\mathsf{i}\in\mathrm{N}} \mathrm{trade}_{\mathsf{i},j} = 1$.

Next, we consider Rule 3. Assume $\mathrm{M}_{sa}, \delta, t \models \text{terminal}$ iff $\delta[t] \neq \bar{\mathrm{w}}$ and for all $j \in \mathrm{G}$, either $\mathrm{M}_{sa}, \delta, t \models \mathrm{trade}_{r,j} = 1$ for some $\mathsf{r} \in \mathrm{N}$ or $\mathrm{M}_{sa}, \delta, t \models \neg\mathrm{bid}_{\mathsf{i},j}$ for all $\mathsf{i} \in \mathrm{N}$. By Rule 2, $\mathrm{M}_{sa}, \delta, t \models \text{terminal}$ iff $\mathrm{M}_{sa}, \delta, t \models \neg\text{initial} \wedge \bigwedge_{j\in\mathrm{G}}(\mathrm{sold}_j \vee \bigwedge_{j\in\mathrm{G}} \neg\mathrm{bid}_{\mathsf{i},j})$.

Now we verify Rule 9. Let $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$. Assume $\mathrm{M}_{sa}, \delta, t \models (\mathrm{does}_\mathsf{i}(p_1, \ldots, p_\mathsf{m}) \wedge p_j \neq 0) \vee (\mathrm{bid}_{\mathsf{i},j} \wedge \text{terminal})$, for some $p_1, \ldots, p_\mathsf{m} \in \mathrm{I}_{\succeq 0}$. We next prove for the two cases. First, assume $\mathrm{M}_{sa}, \delta, t \models \mathrm{bid}_{\mathsf{i},j} \wedge \text{terminal}$. Then $\mathrm{bid}_{\mathsf{i},j} \in \pi_\Phi(\delta[t])$ and $\delta[t] \in \mathrm{T}$. By the update function, $\delta[t+1] = \delta[t]$ and $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{bid}_{\mathsf{i},j}$, *i.e.*, $\mathrm{M}_{sa}, \delta, t \models \bigcirc\mathrm{bid}_{\mathsf{i},j}$. In the second case, assume $\mathrm{does}_\mathsf{i}(p_1, \ldots, p_\mathsf{m}) \wedge p_j \neq 0$. By the update function, $\mathrm{bid}_{\mathsf{i},j} \in \pi_\Phi(\delta[t+1])$ and thus $\mathrm{M}_{sa}, \delta, t \models \bigcirc\mathrm{bid}_{\mathsf{i},j}$.

The remaining rules are verified in a similar way. □

Next, we show that no good can be bought by two different agents, *i.e.*, given any two agents and a good, one of them will have her trade equal to zero. When a good is sold, it will still be sold in the next state.

Proposition 3.1. *For each* $j \in \mathrm{G}$ *and each* $\mathsf{i}, \mathsf{r} \in \mathrm{N}$ *such that* $\mathsf{i} \neq \mathsf{r}$,

1. $\mathrm{M}_{sa} \models \mathrm{trade}_{\mathsf{i},j} = 0 \vee \mathrm{trade}_{\mathsf{r},j} = 0$
2. $\mathrm{M}_{sa} \models \mathrm{sold}_j \rightarrow \bigcirc\mathrm{sold}_j$
3. $\mathrm{M}_{sa} \models \neg\mathrm{sold}_j \rightarrow \mathrm{price} = \mathrm{price}_j$

Proof. Given a path δ in M_{sa}, any stage t of δ and a good type $j \in \mathrm{G}$, let $\mathsf{i}, \mathsf{r} \in \mathrm{N}$, such that $\mathsf{i} \neq \mathsf{r}$. Let us consider Statement 1. If $\delta[t] = \bar{\mathrm{w}}$, then $\mathrm{M}_{sa}, \delta, t \models \mathrm{trade}_{\mathsf{i},j} = 0 \wedge \mathrm{trade}_{\mathsf{r},j} = 0$ (see Rule 1). Otherwise, by the path definition, $\delta[j] = \mathrm{U}(\delta[t-1], \theta(\delta, t-1))$. Let us suppose for the sake of contradiction that $\mathrm{M}_{sa}, \delta, t \not\models \mathrm{trade}_{\mathsf{i},j} = 0 \vee \mathrm{trade}_{\mathsf{r},j} = 0$. Since W construction defines $\mathrm{trade}_{\mathsf{i},j}, \mathrm{trade}_{\mathsf{r},j} \in \{0, 1\}$, we have $\mathrm{M}_{sa}, \delta, t \models \mathrm{trade}_{\mathsf{i},j} = 1 \wedge \mathrm{trade}_{\mathsf{r},j} = 1$. Thus, $\mathrm{M}_{sa}, \delta, t-1 \models \bigcirc\mathrm{trade}_{\mathsf{i},j}$. By Rule 4, $\mathrm{M}_{sa}, \delta, t-1 \models \bigcirc(\mathrm{bid}_{\mathsf{i},j} \wedge \bigwedge_{\mathsf{s}\in\mathrm{N}\setminus\{\mathsf{i}\}} \neg\mathrm{bid}_{\mathsf{s},j})$. Thereby, $\mathrm{M}_{sa}, \delta, t-1 \not\models \bigcirc(\mathrm{bid}_{\mathsf{r},j} \wedge \bigwedge_{\mathsf{s}\in\mathrm{N}\setminus\{\mathsf{r}\}} \neg\mathrm{bid}_{\mathsf{s},j})$ and $\mathrm{M}_{sa}, \delta, t-1 \not\models \bigcirc\mathrm{trade}_{\mathsf{r},j=1}$. Thus, $\mathrm{M}_{sa}, \delta, t \not\models \mathrm{trade}_{\mathsf{r},j} = 1$, which is a contradiction.

For Statement 2. Assume $\mathrm{M}_{sa}, \delta, t \models \mathrm{sold}_j$. Then, $\mathrm{M}_{sa}, \delta, t \models \mathrm{trade}_{\mathsf{i},j} = 1$ for some agent $\mathsf{i} \in \mathrm{N}$. From Rule 6, $\theta_{\mathsf{i}}(\delta, t) = (p_1, \ldots, p_{\mathsf{m}})$ with $p_j = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}_j)$ and $\theta_{\mathsf{r}}(\delta, t) = (p'_1, \ldots, p'_{\mathsf{m}})$ with $p'_j = 0$, for all agent $\mathsf{r} \neq \mathsf{i}$. By Rule 9, we have $\mathrm{M}_{sa}, \delta, t \models \mathrm{bid}_{\mathsf{i},j} \wedge \bigwedge_{\mathsf{r} \in \mathrm{N} \setminus \{\mathsf{i}\}} \neg \mathrm{bid}_{\mathsf{r},j}$. Thus, it follows from Rules 2 and 4 that $\mathrm{M}_{sa}, \delta, t \models \bigcirc \mathrm{trade}_{\mathsf{i},j} = 1$ and $\mathrm{M}_{sa}, \delta, t \models \bigcirc \mathrm{sold}_j$.

For Statement 3, notice all prices in the initial state have the same value (Rule 1) and the current price and the price for unsold items are increased by the same amount in each turn (Rule 7 and 8). Assume $\mathrm{M}_{sa}, \delta, t \models \neg \mathrm{sold}_j$. Since $\mathrm{M}_{sa} \models \mathrm{sold}_j \rightarrow \bigcirc \mathrm{sold}_j$, there is no stage $t' < t$ such that $\mathrm{M}_{sa}, \delta, t' \models \mathrm{sold}_j$ and thus $\mathrm{M}_{sa}, \delta, t \models \mathrm{price} = \mathrm{price}_j$.

□

Each path in M_{sa} reaches a terminal state, and thus the protocol satisfies the termination condition. Furthermore, M_{sa} satisfies playability, that is, there is always a legal action for each agent to take. Thus, Σ_{sa} is well-formed over M_{sa}.

Theorem 3.1. *Σ_{sa} is a well-formed protocol over the ST-model M_{sa}.*

Proof. Since M_{sa} is a model of Σ_{sa} (see Lemma 3.1), we have to show that for each path δ in M_{sa} and each agent $\mathsf{i} \in \mathrm{N}$,

1. δ is a complete path;
2. $\mathrm{M}_{sa} \models \bigvee_{a \in \mathcal{B}} \mathrm{legal}_{\mathsf{i}}(a)$.

Let us start by verifying Statement 1. Remind $\mathsf{start} \in \mathrm{I}_{\succeq 0}$ and $\mathsf{inc} \in \mathrm{I}_{\succ 0}$. Let δ be a path in M_{sa}. In $\delta[0]$, $\pi_{\mathrm{Y}}(\delta[0], \mathrm{price}) = \mathsf{start}$. By the update function, for any stage t, if $\delta[t] \notin \mathrm{T}$, then $\pi_{\mathrm{Y}}(\delta[t+1], \mathrm{price}) = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}) + \mathsf{inc}$.

For the sake of contradiction, let us assume δ is not complete. Let $\mathsf{i} \in \mathrm{N}$ be an agent. By the definition of L, $(p_1, \ldots, p_{\mathsf{m}}) \in \mathrm{L}(\delta[t], \mathsf{i})$, for all $0 \leq p_j < z_{\max} - \mathsf{inc}$ and $j \in \mathrm{G}$, such that either (i) $p_j = 0$ & $\pi_{\mathrm{Y}}(\delta[t], \mathrm{trade}_{\mathsf{i},j}) = 0$, or (ii) $p_j = \mathrm{price} + \mathsf{inc}$ & $\pi_{\mathrm{Y}}(\delta[t], \mathrm{trade}_{r,j}) = 0$ for all $r \in \mathrm{N}$, or (iii) $p_j = \mathrm{price}_j$ & $\pi_{\mathrm{Y}}(\delta[t], \mathrm{trade}_{r,j}) = 0$. Since $\pi_{\mathrm{Y}}(\delta[t+1], \mathrm{price}) > \pi_{\mathrm{Y}}(\delta[t], \mathrm{price})$, there will be a stage $e \geq 0$ in δ, where the condition (ii) will not be true for any $0 \leq p_j < z_{\max} - \mathsf{inc}$. Thus, for each good j, it will be the case that i bids 0 for it or the good was assigned to her (*i.e.*, $\pi_{\mathrm{Y}}(\delta[e], \mathrm{trade}_{\mathsf{i},j}) = 1$). From Rules 3 and 9 in Σ_{sa}, it follows that $\delta[e+1] \in \mathrm{T}$. Thus, δ is a complete path, which is a contradiction.

We now consider Statement 2. Given a path δ in M_{sa} and a stage t in δ, we show that there is a legal action for agent i in $\delta[t]$. For each $j \in \mathrm{G}$, let $p_j = 0$ if $\pi_{\mathrm{Y}}(\delta[t], \mathrm{trade}_{\mathsf{i},j}) = 1$. Otherwise, let $p_j = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}_j)$. By L definition, we have $(p_1, \ldots, p_{\mathsf{m}}) \in \mathrm{L}(\delta[t], \mathsf{i})$. Thus, $\mathrm{M}_{sa}, \delta, t \models \bigvee_{a \in \mathcal{B}} \mathrm{legal}_{\mathsf{i}}(a)$. □

From being a single-side auction where all agents are buyers, it follows that there is no monetary deficit in M_{sa}, but it is not strongly budget-balanced.

Proposition 3.2. $\mathrm{M}_{sa} \models \bigcirc \mathrm{WBB}$ *and* $\mathrm{M}_{sa} \not\models \bigcirc \mathrm{SBB}$

Proof. Suppose a path δ in M_{sa} and a stage t in δ. Note that each trade can be either 0 or 1 and the price for a good is at least 0, *i.e.*, $\pi_{\mathrm{Y}}(\delta[t], \mathrm{trade}_{\mathsf{i},j}) \in \{0,1\}$ and $\pi_{\mathrm{Y}}(\delta[t], \mathrm{price}_j) \in \mathrm{I}_{\geq 0}$. It follows from Rule 10 that $\mathrm{M}_{sa}, \delta, t \models \mathrm{pay}_{\mathsf{i}} \geq 0$ for each agent i and $\mathrm{M}_{sa}, \delta, t \models \mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(\mathrm{pay}_{\mathsf{i}}) \geq 0$. Thus, $\mathrm{M}_{sa} \models \bigcirc\mathrm{WBB}$.

We will prove M_{sa} is not strongly budget-balanced with a counter-example. Given an agent i, let δ be a path in M_{sa} such that $\theta_{\mathsf{i}}(\delta, 0) = (\mathsf{start}, 0, \ldots, 0)$ and $\theta_{\mathsf{s}}(\delta, 0) = (0, \ldots, 0)$ for each agent $\mathsf{s} \neq \mathsf{i}$. Since $(\mathsf{start}, 0, \ldots, 0) \in \mathrm{L}(\delta[0], \mathsf{i})$ and $(0, \ldots, 0) \in \mathrm{L}(\mathrm{w}, \mathsf{s})$, there exists such path in M_{sa}. Since $\delta[0] \notin \mathrm{T}$, $\pi_{\mathrm{Y}}(\delta[0], \mathrm{price}) = \pi_{\mathrm{Y}}(\delta[0], \mathrm{price}_j) = \mathsf{start}$ and $\mathrm{sold}_j \notin \pi_{\Phi}(\delta[0])$ for each j, it follows from Rules 7 and 8, that all prices are increased by the constant $\mathsf{inc} > 0$, that is $\mathrm{M}_{sa}, \delta, 1 \models \mathrm{price} = \mathrm{sum}(\mathsf{start}, \mathsf{inc}) \wedge \bigwedge_{j\in\mathrm{G}} \mathrm{price}_j = \mathrm{sum}(\mathsf{start}, \mathsf{inc})$. By Rule 9, we have that agent i is only bidding for the good 1, that is, $\mathrm{M}_{sa}, \delta, 1 \models \mathrm{bid}_{\mathsf{i},1} \wedge \bigwedge_{j\in\mathrm{G}\setminus\{1\}} \neg\mathrm{bid}_{\mathsf{i},j}$. All other agents are not bidding for any good, *i.e.*, $\mathrm{M}_{sa}, \delta, 1 \models \bigwedge_{j\in\mathrm{G}} \neg\mathrm{bid}_{\mathsf{s},j}$, for each $\mathsf{s} \neq \mathsf{i}$. From Rules 4 and 10, we have $\mathrm{M}_{sa}, \delta, 1 \models \mathrm{trade}_{\mathsf{i},1} = 1 \wedge \mathrm{pay}_{\mathsf{i}} = \mathrm{price}_1 \wedge \bigwedge_{\mathsf{s}\in\mathrm{N}\setminus\{\mathsf{i}\}} \mathrm{p}_{\mathsf{s}} = 0$. Since $\pi_{\mathrm{Y}}(\delta[1], \mathrm{price}_1) > 0$, we have $\mathrm{M}_{sa}, \delta, 1 \not\models \mathrm{sum}_{\mathsf{s}\in\mathrm{N}}(\mathrm{pay}_{\mathsf{s}}) = 0$ and $\mathrm{M}_{sa} \not\models \bigcirc\mathrm{SBB}$.

□

The simultaneous ascending auction is only efficient on states preceding the terminal one.

Proposition 3.3. *Given a joint action* $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$*,*

1. $\mathrm{M}_{sa} \not\models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{EF}(\boldsymbol{\beta})$
2. $\mathrm{M}_{sa} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc(\mathrm{terminal} \rightarrow \mathrm{EF}(\boldsymbol{\beta}))$

Proof. Given a path δ in M_{sa}, we first consider Statement 1. We prove M_{sa} is not efficient with a counter-example. Our purpose is to show that there exists a joint action $\boldsymbol{\beta}$ such that $\mathrm{M}_{sa}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta}) \wedge \neg \bigcirc \mathrm{EF}(\boldsymbol{\beta})$. That is, $\mathrm{M}_{sa}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta}) \wedge \neg \bigcirc \mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_{\boldsymbol{j}})_{j\in\mathrm{G}})) = \max_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}}(\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda})))$.

As we consider the initial state of δ, $\mathrm{M}_{sa}, \delta, 0 \models \mathrm{price} = \mathsf{start} \wedge \bigwedge_{j\in\mathrm{G}} \mathrm{price}_j = \mathsf{start} \wedge \bigwedge_{\mathsf{i}\in\mathrm{N}}(\neg\mathrm{bid}_{\mathsf{i},j} \wedge \mathrm{trade}_{\mathsf{i},j} = 0)$. Thus, no good was sold, that is, $\mathrm{M}_{sa}, \delta, 0 \models \bigwedge_{j\in\mathrm{G}} \neg\mathrm{sold}_j$. Let i and r be two distinct agents in N with $\mathsf{i} \neq \mathsf{r}$. From the definition of L, $\mathrm{L}(\delta[0], \mathsf{i}) = \mathrm{L}(\delta[0], \mathsf{r})$ and $(p_1, \ldots, p_{\mathsf{m}}) \in \mathrm{L}(\delta[0], \mathsf{i})$ if $p_j = 0$ or $p_j = \mathrm{sum}(\mathsf{start}, \mathsf{inc})$ for each good j.

Let us assume $\mathrm{M}_{sa}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta})$ for $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$ such that $\beta_{\mathsf{i}} = \beta_{\mathsf{r}} = (\mathsf{start} + \mathsf{inc}, 0, \ldots, 0)$ and $\beta_{\mathsf{s}} = (0, \ldots, 0)$ for each $\mathsf{s} \in \mathrm{N} \setminus \{\mathsf{i}, \mathsf{r}\}$. Since $\beta_{\mathsf{i}} \in \mathrm{L}(\delta[0], \mathsf{i})$, $\beta_{\mathsf{r}} \in \mathrm{L}(\delta[0], \mathsf{r})$ and $\beta_{\mathsf{s}} \in \mathrm{L}(\delta[0], \mathsf{s})$, there exists such path in M_{sa}. Thus, $\mathrm{M}_{sa}, \delta, 1 \models \mathrm{bid}_{\mathsf{i},1} \wedge \mathrm{bid}_{\mathsf{r},1}$. Notice $\mathrm{bid}_{\mathsf{i}',j}$ does not hold in $\delta[1]$ for any other pair $(\mathsf{i}', j) \neq (\mathsf{i}, 1)$ and $(\mathsf{i}', j) \neq (\mathsf{s}, 1)$. From Rules 4 and 5, we have $\mathrm{M}_{sa}, \delta, 1 \models \bigwedge_{\mathsf{i}'\in\mathrm{N}, j\in\mathrm{G}} \mathrm{trade}_{\mathsf{i}',j} = 0$.

Recall the valuation of each agent i' is $v'_{\mathsf{i}}((p_1, \ldots, p_{\mathsf{m}}), \boldsymbol{\lambda}) = \sum_{j\in\mathrm{G}} \lambda_{\mathsf{i}',j} \cdot p_j$ for a trade $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$ and a bid $(p_1, \ldots, p_{\mathsf{m}}) \in \mathcal{B}$. Since each trade has the value 0 in $\delta[1]$, it follows that $\mathrm{M}_{sa}, \delta, 1 \models \mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_{\boldsymbol{j}})_{j\in\mathrm{G}})) = 0$.

However, let $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$ be a trade such that $\lambda_{\mathsf{i},1} = 1$ and $\lambda_{\mathsf{i}',j} = 0$ for all other pair $(\mathsf{i}', j) \neq (\mathsf{i}, 1)$. It is easy to see that $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$. The value of this trade for i is $v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda}) =$

$\mathsf{start} + \mathsf{inc}$ and $v'_{\mathsf{i}}(\beta'_{\mathsf{i}}, \boldsymbol{\lambda}) = 0$ for each $\mathsf{i}' \neq \mathsf{i}$. Thus, we have $\sum_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda})) = \mathsf{start} + \mathsf{inc}$, $\mathrm{M}_{sa}, \delta, 1 \models \neg\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_{j})_{j\in\mathrm{G}})) = \max_{\boldsymbol{\lambda}\in\Lambda}(\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda})))$ and $\mathrm{M}_{sa}, \delta, 0 \models \neg \bigcirc \mathrm{EF}(\boldsymbol{\beta})$.

For Statement 2, let $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$ be a joint action and $t \geq 0$ be a stage of δ. From the definition of M_{sa} we have that each trade can be only 0 or 1. That is, if $\boldsymbol{\lambda} \in \Lambda$ then $\lambda_{\mathsf{i},j} \in \{0, 1\}$ for each agent i and good j. Assume $\mathrm{M}_{sa} \models \mathrm{does}(\boldsymbol{\beta})$ and $\mathrm{M}_{sa}, \delta, t \models \bigcirc\mathrm{terminal}$. We intend to show that $\mathrm{M}_{sa} \models \bigcirc\mathrm{EF}(\boldsymbol{\beta})$, *i.e.*, $\mathrm{M}_{sa}, \delta, t \models \bigcirc\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_{j})_{j\in\mathrm{G}})) = \max_{\boldsymbol{\lambda}\in\Lambda}(\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda}))))$.

We focus on the case where $\delta[t] \notin \mathrm{T}$. Let i be an agent in N and let $(p_1, \ldots, p_{\mathsf{m}})$ denote i's action in $\boldsymbol{\beta}$. Recall function $v_{\mathsf{i}}((p_1, \ldots, p_{\mathsf{m}}), \boldsymbol{\lambda}) = \sum_{j\in\mathrm{G}} p_j \cdot \lambda_{\mathsf{i},j}$, for a trade $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$. Since the value for each good in v_{i} depends only on its reported value in i's bid and its trade for i, we show that the part of $v_{\mathsf{i}}((p_1, \ldots, p_{\mathsf{m}}), \boldsymbol{\lambda})$ corresponding to each good j (*i.e.*, $p_j \cdot \lambda_{\mathsf{i},j}$) is maximized when $\lambda_{\mathsf{i},j} = \pi_{\mathrm{Y}}(\delta[t], \mathrm{trade}_{\mathsf{i},j})$.

From the definition of $\mathcal{B}$, we have that the reported value for j in i's bid is at least zero, *i.e.*, $p_j \geq 0$. We check for two cases:

1. If $p_j = 0$, then $\lambda_{\mathsf{i}j} \cdot p_j = 0$, for any trade $\boldsymbol{\lambda} \in \Lambda$. Thus, $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{times}(p_j, \mathrm{trade}_{\mathsf{i},j}) = \max(\mathrm{times}(p_j, 0), \mathrm{times}(p_j, 1))$;

2. If $p_j > 0$, then $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{bid}_{\mathsf{i},j}$. Since $\delta[t+1] \in \mathrm{T}$, it should be the case that $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{sold}_{j}$. From Rule 2, we know the trade for good j is one for some agent, *i.e.*, $\mathrm{M}_{sa}, \delta, t+1 \models \bigvee_{\mathsf{r}\in\mathrm{N}} \mathrm{trade}_{\mathsf{r},j} = 1$. Because $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{bid}_{\mathsf{i},j}$, it should be the case that the agent who have a trade for j is i (see Rules 4 and 5), that is, $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{trade}_{\mathsf{i},j} = 1$. Hence, $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{times}(p_j, 1) = \max(\mathrm{times}(p_j, 0), \mathrm{times}(p_j, 1))$.

It follows that

$$\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_{j})_{j\in\mathrm{G}})) = \max_{\boldsymbol{\lambda}\in\Lambda}(\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda}))))$$

or simply, $\mathrm{M}_{sa}, \delta, t \models \bigcirc ef(\boldsymbol{\beta})$.

The case where $\delta[t] \in \mathrm{T}$, follows from the loop on the path definition. □

The auction described by M_{sa} is individually rational, since agents pay at most their bids.

Theorem 3.2. *Given a joint action* $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$, $\mathrm{M}_{sa} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{IR}(\boldsymbol{\beta})$

Proof. Given a path δ in M_{sa} and a stage t, assume $\mathrm{M}_{sa}, \delta, t \models \mathrm{does}(\boldsymbol{\beta})$ for some $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$. We consider first the case where $\delta[t] \notin \mathrm{T}$. Let $\mathsf{i} \in \mathrm{N}$ be an agent and $(p_1, \ldots, p_{\mathsf{m}})$ denote the bid of i in the joint action $\boldsymbol{\beta}$. Let us consider the good $j \in \mathrm{G}$. We denote by $pr_j = \pi_{\mathrm{Y}}(\delta[t+1], \mathrm{price}_j)$ the price of good j in $\delta[t+1]$ and by $\boldsymbol{\lambda} = (\pi_{\mathrm{Y}}(\delta[t+1], \mathrm{trade}_{\mathsf{s},j'}))_{j'\in\mathrm{G},\mathsf{s}\in\mathrm{N}}$ the trade in $\delta[t+1]$. Recall function $v_{\mathsf{i}}((p_1, \ldots, p_{\mathsf{m}}), \boldsymbol{\lambda}) = \sum_{j\in\mathrm{G}} \lambda_{\mathsf{i},j} \cdot p_j$. Similarly, by Rule 10, we have that the payment for agent i in $\delta[t+1]$ is $\pi_{\mathrm{Y}}(\delta[t+1], \mathrm{pay}_{\mathsf{i}}) = \sum_{j\in\mathrm{G}} pr_j \cdot \lambda_{\mathsf{i},j}$.

We have $\theta_{\mathsf{i}}(\delta, t) = (p_1, \ldots, p_{\mathsf{m}})$. Notice $\mathsf{inc}, p_j \in \mathrm{I}_{\geq 0}$ by the definition of $\mathcal{S}_{sa}$. According with the action legality (Rule 6), the value of p_j can be either zero, the price of j in $\delta[t]$ or the current price incremented by inc. For each of the three cases, we show that the part of i's payment corresponding to j is equal to the part of $v_{\mathsf{i}}((p_1, \ldots, p_{\mathsf{m}}), \boldsymbol{\lambda})$ corresponding to j:

- If $p_j = 0$, agent i chosen to not bid for good j. By Rules 5 and 9, $\mathrm{M}_{sa}, \delta, t+1 \models \neg\mathrm{bid}_{\mathsf{i},j} \wedge \mathrm{trade}_{\mathsf{i},j} = 0$. Thus, we have $\lambda_{\mathsf{i},j} \cdot p_j = pr_j \cdot \lambda_{\mathsf{i},j} = 0$.
- If $p_j = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}_j)$, by the definition of L, it must be the case that the good was already sold to agent i, that is $\mathrm{M}_{sa}, \delta, t \models \mathrm{trade}_{\mathsf{i},j} = 1 \wedge \mathrm{sold}_j$. Since good j is sold, Rule 6 ensure other agents can only bid the value 0 for j, and thus, $\mathrm{M}_{sa}, \delta, t+1 \models \bigwedge_{r \in \mathrm{N} \setminus \{\mathsf{i}\}} \neg\mathrm{bid}_{r,j}$. By Rules 4 and 9, we have $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{bid}_{\mathsf{i},j} \wedge \mathrm{trade}_{\mathsf{i},j} = 1$. Since good j is sold, its price in $\delta[t]$ is the same as in $\delta[t]$ (Rule 8), that is $\mathrm{M}_{sa}, \delta, t \models \mathrm{price}_j = p_j$. Thus, we have $\lambda_{\mathsf{i},j} \cdot p_j = pr_j \cdot \lambda_{\mathsf{i},j} = pr_j$.
- Let $prevprice = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price})$. If $p_j = prevprice + \mathsf{inc}$, then $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{bid}_{\mathsf{i},j}$. From the legality definition, it should be the case that j is not sold, that is $\mathrm{M}_{sa}, \delta, t \models \neg\mathrm{sold}_j \wedge \mathrm{price}_j = \mathrm{sum}(prevprice_j, \mathsf{inc})$. From Statement 3 of Proposition 3.1, we have $\mathrm{M}_{sa}, \delta, t \models \mathrm{price} = \mathrm{price}_j$. The value of $\mathrm{trade}_{\mathsf{i},j}$ depends on the actions of other agents in $\delta[t]$. From Rules 5 and 4, we have $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{trade}_{\mathsf{i},j} = 0 \vee \mathrm{trade}_{\mathsf{i},j} = 1$. In the first case, $\lambda_{\mathsf{i},j} \cdot p_j = pr_j \cdot \lambda_{\mathsf{i},j} = 0$. Otherwise, the trade has the value 1 and $\lambda_{\mathsf{i},j} \cdot p_j = pr_j \cdot \lambda_{\mathsf{i},j} = pr_j$, since $p_j = pr_j = prevprice + \mathsf{inc}$.

It follows that $\mathrm{M}_{sa}, \delta, t+1 \models v_{\mathsf{r}}(\beta, (\mathrm{trade}_{\boldsymbol{j}})_{j \in \mathrm{G}}) = \mathrm{pay}_{\mathsf{r}}$ and $\mathrm{M}_{sa}, \delta, t+1 \models \mathrm{sub}(v_{\mathsf{r}}(\beta, (\mathrm{trade}_{\boldsymbol{j}})_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{r}}) \geq 0$. Thus, $\mathrm{M}_{sa}, \delta, t \models \bigcirc\mathrm{IR}(\boldsymbol{\beta})$.

If $\delta[t] \in \mathrm{T}$, the loop in the path definition ensures $\mathrm{M}_{sa}, \delta, t \models \bigcirc\mathrm{IR}(\boldsymbol{\beta})$ if and only if $\mathrm{M}_{sa}, \delta, t \models \mathrm{IR}(\boldsymbol{\beta})$. □

Under the assumption that each stage in M_{sa} is a (direct) mechanism for which the legality set represents the agents' preference spaces, the SAA is strategyproof. As it is only legal to accept or decline to raise the current price for unsold goods (represented by bidding the value 0 or price + inc), there is no utility improvement if the agent accepts when she would prefer to decline (and vice-versa). When a good is sold for an agent, there is only one value that is legal to bid for the good bought, and thus the agent cannot strategize.

Proposition 3.4. M_{sa} *is strategyproof.*

Proof. Given a path δ in M_{sa} and a stage $t \geq 0$ in δ, such that $\mathrm{L}(\delta[t], \mathsf{i}) \approx_{\mathsf{i}} V_{\mathsf{i}}$. Let $\boldsymbol{\vartheta} \in \prod_{\mathsf{i} \in \mathrm{N}} V_{\mathsf{i}}$ be a preference profile and $\delta_{\boldsymbol{\vartheta}}$ denote a path such that $\delta[0, t] = \delta_{\boldsymbol{\vartheta}}[0, t]$ and $\theta(\delta_{\boldsymbol{\vartheta}}, t) = \boldsymbol{\beta}_{\boldsymbol{\vartheta}}$. We let $\vartheta'_{\mathsf{i}} \in V_{\mathsf{i}}$ denote a preference of agent i, $\boldsymbol{\vartheta'} = (\vartheta'_{\mathsf{i}}, \vartheta_{-\mathsf{i}})$ and $\delta_{\boldsymbol{\vartheta'}}$ be a path such that $\delta[0, t] = \delta_{\boldsymbol{\vartheta'}}[0, t]$, $\theta_{\mathsf{i}}(\delta_{\boldsymbol{\vartheta'}}, t) = \beta_{\vartheta'_{\mathsf{i}}}$ and $\theta_{\mathsf{r}}(\delta_{\boldsymbol{\vartheta'}}, t) = \beta_{\vartheta_{\mathsf{r}}}$ for each agent $\mathsf{r} \neq \mathsf{i}$.

In the stage t of $\delta_{\boldsymbol{\vartheta}}$, the agents report their (truthful) preferences $\boldsymbol{\vartheta}$, *i.e.*, $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}}, t \models \mathrm{does}(\boldsymbol{\beta}_{\boldsymbol{\vartheta}})$. On the other hand, in the stage t of $\delta_{\boldsymbol{\vartheta}'}$, the agent i reports her (untruthful) preference ϑ'_{i} and each agent $\mathsf{r} \neq \mathsf{i}$ reports ϑ_{r}, *i.e.*, $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \mathrm{does}_{\mathsf{i}}(\beta_{\vartheta'_{\mathsf{r}}}) \wedge \bigwedge_{\mathsf{r} \in \mathrm{N} \setminus \{\mathsf{i}\}} \mathrm{does}_{\mathsf{r}}(\beta_{\vartheta_{\mathsf{r}}})$.

For some $x \in \mathrm{I}$, we have to show $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}}, t \models \bigcirc\mathrm{sub}(v_{\mathsf{i}}(\beta_{\vartheta_{\mathsf{i}}}, (\mathrm{trade}_{\boldsymbol{j}})_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) = x$ and $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{sub}(v_{\mathsf{i}}(\beta_{\vartheta_{\mathsf{i}}}, (\mathrm{trade}_{\boldsymbol{j}})_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) \leq x$.

That is, agent i's utility in $\delta_{\boldsymbol{\vartheta}'}$ is not better than in $\delta_{\boldsymbol{\vartheta}}$. Remind the legal actions in $\delta[t]$ represent her preference space, *i.e.*, $\mathrm{L}(\delta[t], \mathsf{i}) \approx_{\mathsf{i}} V_{\mathsf{i}}$. Notice the value of a bid $(p_1, \ldots, p_{\mathsf{m}})$ given a trade $\boldsymbol{\lambda}$ is $v_{\mathsf{i}}((p_1, \ldots, p_{\mathsf{m}}), \boldsymbol{\lambda}) = \sum_{j \in \mathrm{G}} \lambda_{\mathsf{i},j} \cdot p_j$. Similarly, the payment in $\delta[t]$ is $\pi_{\mathrm{Y}}(\delta[t], \mathrm{pay}_{\mathsf{i}}) = \sum_{j \in \mathrm{G}} \pi_{\mathrm{Y}}(\delta[t], \mathrm{trade}_{\mathsf{i},j}) \cdot \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}_j)$. Since there is no dependence among goods in v_{i} and in i's payment, we consider the part of i's utility corresponding to j in $\delta_{\boldsymbol{\vartheta}}$ and $\delta_{\boldsymbol{\vartheta}'}$. Thus, we need to show that

(i) $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) = x_j$

(ii) $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) \leq x_j$

for some $x_j \in \mathrm{I}$ and each good $j \in \mathrm{G}$.

We denote $\beta_{\vartheta_{\mathsf{i}}} = (p_1, \ldots, p_{\mathsf{m}})$ and $\beta_{\vartheta'_{\mathsf{i}}} = (p'_1, \ldots, p'_{\mathsf{m}})$. According to the legality definition, the values of p_j and p'_j can be either the price of j in $\delta[t]$, zero, or the price in $\delta[t]$ incremented by $\mathsf{inc} > 0$. Let us consider each case:

- Assume $p_j = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}_j)$ iff $\mathrm{M}_{sa}, \delta, t \models \mathrm{trade}_{\mathsf{i},j} = 1$ (w.r.t the definition of L). That is, $p_j = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}_j)$ when good j was already bought by agent i. Thus, it is not legal for i to bid any other value for j, *i.e.*, $(p'_1, \ldots, p'_j) \in \mathrm{L}(\delta[t], \mathsf{i})$ iff $p_j = p'_j$. Thereby, it is easy to see that $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) = p_j$ and also $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) = p_j$.

- Assume agent i declines to raise her bid for good j, that is, $p_j = 0$. By Rules 5 and 9, we have $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}}, t \models \bigcirc\mathrm{trade}_{\mathsf{i},j} = 0$ and $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) = 0$. By the legality definition, it should be the case that $\mathrm{M}_{sa}, \delta, t \models \neg\mathrm{trade}_{\mathsf{i},j} = 0$ and p'_j is either 0 or $\pi_{\Phi}(\delta[t], \mathrm{price}) + \mathsf{inc}$. If $p'_j = 0$, then $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) = 0$. Otherwise, the value of $\mathrm{trade}_{\mathsf{i},j}$ in $\delta_{\boldsymbol{\vartheta}'}[t+1]$ will be either 0 or 1, depending on the joint bid $\boldsymbol{\beta}_{\boldsymbol{\vartheta}'}$.

  - If $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{trade}_{\mathsf{i},j} = 1$, then $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) = \mathrm{sub}(0, \mathrm{price}_j)$. Since $\pi_{\mathrm{Y}}(\delta_{\boldsymbol{\vartheta}'}[t+1], \mathrm{price}_j) \geq 0$, we have $\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) \leq 0$.
  - Otherwise, the trade for good j is zero (*i.e.*, $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{trade}_{\mathsf{i},j} = 0$) and $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{sub}(\mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, p_j), \mathrm{times}(\mathrm{trade}_{\mathsf{i},j}, \mathrm{price}_j)) = 0$.

- The proof for the case $p_j = \pi_{\mathrm{Y}}(\delta[t], \mathrm{price}) + \mathsf{inc}$ is similar to the previous case.

It follows that $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}}, t \models \bigcirc\mathrm{sub}(v_{\mathrm{i}}(\beta_{\vartheta_{\mathrm{i}}}, (\mathrm{trade}_{\boldsymbol{j}})_{j\in\mathrm{G}}), \mathrm{pay}_{\mathrm{i}}) = x$ and $\mathrm{M}_{sa}, \delta_{\boldsymbol{\vartheta}'}, t \models \bigcirc\mathrm{sub}(v_{\mathrm{i}}(\beta_{\vartheta_{\mathrm{i}}}, (\mathrm{trade}_{\boldsymbol{j}})_{j\in\mathrm{G}}), \mathrm{pay}_{\mathrm{i}}) \leq x$, where $x \in \mathrm{I}$. □

We conclude the section by discussing variants of non-combinatorial auctions. When the number of good types is one (that is, $|\mathrm{G}| = 1$), Σ_{sa} corresponds to the Japanese-English auction. For representing the standard variant of the English auction, one should define the legality rule such that agents are also allowed to bid any price above the current price. Furthermore, the price in the next state should be updated according to the highest bid in the current one. The Dutch auction is also similar to Σ_{sa}. The key difference is that in the Dutch auction the bidding value should decrease at each round until at least one agent accepts to pay the current price. As we saw in this section, with the ADL description of a given auction, we are able to formally analyze it, both in relation to domain-specific properties, well-formedness of its protocol and from a mechanism design perspective.

3.2 Combinatorial Exchange

We now consider two protocols for combinatorial exchange: a one-shot protocol and a multi-stage variant. We consider the setting with multiple goods and multiple copies of each good. Agents hold an initial allocation of goods and can trade items with each other. Remind a trade denotes the number of goods being exchanged among the agents in a state. With *allocation* we refer to the number of goods the agents initially have.

Both protocols use the Tree-Based Bidding Language (TBBL) for defining its action set. TBBL (Parkes et al., 2005; Lubin et al., 2008) is a language designed for Combinatorial Exchange. It allows to represent buyers and sellers demands in the same structure. We adopt TBBL as it is a highly expressive and compact language. TBBL is general enough to represent any kind of utility function (full expressivity), like OR-like bidding languages (Boutilier and Hoos, 2001). It is even more expressive in the sense that it is able to mix preferences for buying and selling bundles in the same framework. In relation to which kind of utility functions this framework is able to represent concisely, Cavallo et al. (2005) compares TBBL with XOR and OR* bidding languages with this respect, and shows that TBBL is more compact, in the sense that there are valuation functions that admit an exponentially larger representation in these latter languages than in TBBL.

3.2.1 Tree-Based Bidding Language

The bidding language we present in this section, denoted $\mathcal{L}_{\mathsf{TBBL}}$, only differs from the original definition of TBBL in the fact that we assume all language components and related optimization problems are bounded by I.

Definition 3.1. A formula in $\mathcal{L}_{\mathsf{TBBL}}$ is called a *bid-tree* (or simply a *bid*) and is

generated by the following BNF:

$$\beta ::= \langle z, j, z\rangle \mid \mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, z)$$

where $\bar{\beta} ::= \bar{\beta}, \beta \mid \beta$ is a nonempty *bid list*, $j \in \mathrm{G}$, $z \in \mathrm{I}$, $\mathrm{y} \leq \mathrm{x}$ and $\mathrm{y}, \mathrm{x} \in \mathrm{I}_{\geq 0}$.

A bid in the form $\langle \mathsf{q}, j, \mathsf{v}\rangle$ is called a leaf and represents that the agent is willing to buy (or sell) q units of the good j and pay (or receive) v. The *interval-choose* (IC) operator defines a range on the number of child nodes that must be satisfied. Thus, a bid $\mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, \mathsf{v})$ indicates the agent is willing to pay (or receive) v for the satisfaction of at least y and at most x of the children nodes $\bar{\beta}$. The IC operator can express logical connectors. For instance, $\mathrm{IC}^{1}_{1}(\bar{\beta}, \mathsf{v})$ is equivalent to the XOR operator applied to the bids in the list $\bar{\beta}$. Let $\mathsf{z} = |\bar{\beta}|$ (*i.e.*, the list size), $\mathrm{IC}^{\mathsf{z}}_{\mathsf{z}}(\bar{\beta}, \mathsf{v})$ is equivalent to an AND operator and $\mathrm{IC}^{1}_{\mathsf{z}}(\bar{\beta}, \mathsf{v})$ is equivalent to an OR operator. For simplicity, we hereafter use the corresponding shortcuts $\mathrm{XOR}(\bar{\beta}, \mathsf{v})$, $\mathrm{AND}(\bar{\beta}, \mathsf{v})$ and $\mathrm{OR}(\bar{\beta}, \mathsf{v})$.

For instance, in Figure 3.3, agent i reports her willingness to buy 1 unit of a paying 2€ or 2 units of a for 5€ or to sell 1 unit of b receiving 2€. Agent s bids an exclusive disjunction for either (i) to sell one unit of a and receive 3€; or (ii) to sell 2 units of a receiving 4€ and to buy one unit of b paying 2€. The node representing the condition (ii) has an additional value of 1.

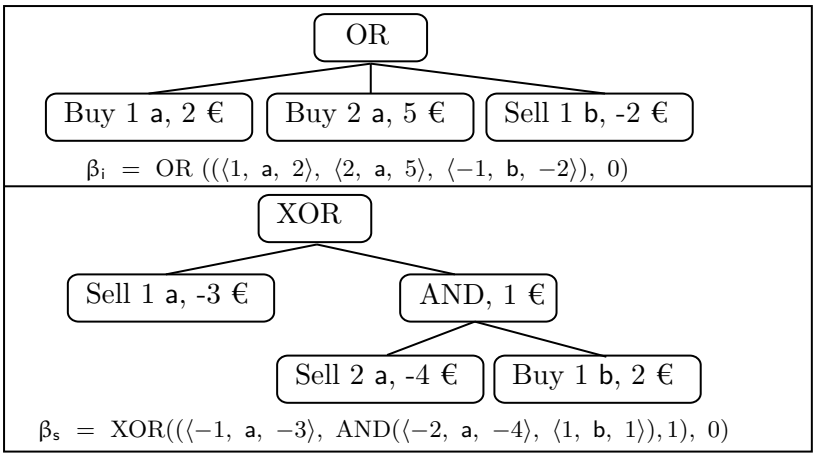

Figure 3.3: Examples of tree-bids β_{i} and β_{s} reported by agents i and s, resp.

Hereafter, we introduce some extra notations to characterize solutions and winners. Let $\beta_{\mathsf{i}} \in \mathcal{L}_{\mathsf{TBBL}}$ be a bid-tree from bidder i, the set $\mathrm{Node}(\beta_{\mathsf{i}})$ denotes all nodes in the tree β_{i}, that is, all its inner bids, including β_{i} itself. Formally, if β_{i} is in the form $\langle \mathsf{q}, j, \mathsf{v}\rangle$, then $\mathrm{Node}(\beta_{\mathsf{i}}) = \{\beta_{\mathsf{i}}\}$. Otherwise, β_{i} is in the form $\mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, \mathsf{v}')$ and $\mathrm{Node}(\beta_{\mathsf{i}}) = \{\beta_{\mathsf{i}}\} \cup \mathrm{Node}(\bar{\beta}_1) \cup \cdots \cup \mathrm{Node}(\bar{\beta}_{\mathsf{z}})$, where $\mathsf{z} = |\bar{\beta}|$ and $\bar{\beta}_k$ is the k-th element of $\bar{\beta}$.

Let $\alpha \in \mathrm{Node}(\beta_{\mathsf{i}})$, the set $\mathrm{Child}(\alpha) \subset \mathrm{Node}(\beta_{\mathsf{i}})$ denotes the children of node α. If α is in the form $\mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, \mathsf{v})$, then $\mathrm{Child}(\alpha) = \{\bar{\beta}_1, \ldots, \bar{\beta}_{\mathsf{z}}\}$, where $\mathsf{z} = |\bar{\beta}|$.

Otherwise, $\text{Child}(\alpha) = \{\}$. The leaves of a bid-tree β_i are denoted by $\text{Leaf}(\beta_i) = \{\langle q, j, v\rangle \in \text{Node}(\beta_i) : q \in I, v \in I \;\&\; j \in G\}$. The value specified at node α is denoted by $b_i(\alpha) \in I$. If α is in the form $\langle q, j, v\rangle$, then $b_i(\alpha) = v$. Otherwise, α is in the form $\text{IC}^x_y(\bar{\beta}, v')$ and $b_i(\alpha) = v'$. Finally, the quantity of units of the good j specified at a leaf $\alpha = \langle q, j, v\rangle$ is denoted by $q(\alpha, j) = q$. For any other $j' \neq j \in G$, $q(\beta, j') = 0$. For any node $\alpha \notin \text{Leaf}(\beta_i)$ and $j \in G$, $q(\alpha, j) = 0$. If α is not a leaf (*i.e.*, $\alpha \in \text{Node}(\beta_i) \setminus \text{Leaf}(\beta_i)$), then it is in the form $\text{IC}^x_y(\bar{\beta})$ and we denote by x_β and y_β the interval-choose constraints x and y, respectively.

3.2.1.1 Trade value and valid solutions

Given a bid-tree β_i from agent i, the value of a trade $\boldsymbol{\lambda} \in I^{nm}$ is defined as the sum of the values in all satisfied nodes, where the set of satisfied nodes is chosen to provide the *maximal* total value. Let $\text{sat}_i(\alpha) \in \{0, 1\}$ denote whether a node $\alpha \in \text{Node}(\beta_i)$ is satisfied and $\text{sat}_i = \{\alpha : \text{sat}_i(\alpha) = 1, \text{ for all } \alpha \in \text{Node}(\beta_i)\}$ denote the nodes satisfied in a solution.

A solution sat_i is valid for a tree β_i and trade λ_i, written $\text{sat}_i \in \text{valid}(\beta_i, \lambda_i)$ if the following rules R1 and R2 hold (Lubin et al., 2008):

$$x_\beta \text{sat}_i(\alpha) \leq \sum_{\gamma \in \text{Child}(\alpha)} \text{sat}_i(\gamma) \leq y_\beta \text{sat}_i(\alpha)$$

$$\forall \alpha \in \text{Node}(\beta_i) \setminus \text{Leaf}(\beta_i) \tag{R1}$$

$$\sum_{\alpha \in \text{Leaf}(\beta_i)} q_i(\alpha, j)\text{sat}_i(\alpha) \leq \lambda_{i,j}, \forall j \in G \tag{R2}$$

Rule R1 ensures that no more and no less than the appropriate number of children are satisfied for any node that is satisfied. Rule R2 requires that the total increase in quantity of each item across all satisfied leaves is no greater than the total number of units traded.

The total value of trade $\boldsymbol{\lambda} \in I^{nm}$ for agent i, given her bid β_i, is defined as the solution to the following problem:

$$v_i(\beta_i, \boldsymbol{\lambda}) = \max_{\text{sat}_i} \sum_{\beta \in \text{Node}(\beta_i)} b_i(\beta) \cdot \text{sat}_i(\beta)$$

$$\text{s.t. rules } R1, R2 \text{ hold}$$

3.2.1.2 Winner determination

Given an auction signature, the bid-trees $\boldsymbol{\beta} = (\beta_i)_{i \in N}$ and an allocation $X = (x_{i,j})_{j \in G, i \in N}$, where $\beta_i \in \mathcal{L}_{\text{TBBL}}$ denotes the bid from agent $i \in N$ and $x_{i,j} \in I_{\succeq 0}$ represents how many copies of j agent i initially holds.

Definition 3.2. The winner determination (WD) defines a pair $(\boldsymbol{\lambda}, sat)$ obtained

by the solution to the following mixed-integer program (Lubin et al., 2008):

$$\mathrm{WD}(\boldsymbol{\beta}, X) : \max_{\boldsymbol{\lambda},\mathrm{sat}} \sum_{\mathsf{i}\in\mathrm{N}} \sum_{\beta\in\mathrm{Node}(\beta_\mathsf{i})} b_\mathsf{i}(\beta)\cdot \mathrm{sat}_\mathsf{i}(\beta)$$

$$\text{s.t. } \lambda_{\mathsf{i},j} + x_{\mathsf{i},j} \geq 0, \forall \mathsf{i}\in\mathrm{N}, j\in\mathrm{G} \tag{C1}$$

$$\sum_{\mathsf{i}\in\mathrm{N}} \lambda_{\mathsf{i},j} \leq 0, \forall j\in\mathrm{G} \tag{C2}$$

$$\mathrm{sat}_\mathsf{i} \in \mathrm{valid}(\beta_\mathsf{i}, \lambda_\mathsf{i}), \forall \mathsf{i}\in\mathrm{N} \tag{C3}$$

$$\mathrm{sat}_\mathsf{i}(\beta) \in \{0,1\}, \lambda_{\mathsf{i},j}\in\mathrm{I} \tag{C4}$$

where $\mathrm{sat} = (\mathrm{sat}_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$. Constraint C1 ensures that the trade $\boldsymbol{\lambda}$ is *feasible* given X, that is, no agent sells more items then she initially hold. Constraint C2 provides free disposal and allows trades to sell more items than are purchased (but not vice-versa). Constraint C3 ensures that each trade for an agent i is valid given her bid-tree. Constraint C4 defines the range for trades and node satisfaction. We denote by $\mathrm{WD}_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, X)$ a function that obtains the trade $\boldsymbol{\lambda}$ in the solution $\mathrm{WD}(\boldsymbol{\beta}, X) = (\boldsymbol{\lambda}, \mathrm{sat})$. Similarly, $\mathrm{WD}_{\lambda_{\mathsf{i},j}}(\boldsymbol{\beta}, X)$ captures the number of units of j traded by agent i in $\mathrm{WD}_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, X)$.

If there are two or more solutions for $\mathrm{WD}(\boldsymbol{\beta}, X)$, the trade $\mathrm{WD}_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, X)$ will be chosen w.r.t. some total order among the elements of I^{nm}. This tie-breaking order is omitted to avoid overloading the notation. In the examples, we assume this order is compatible with the Pareto dominance relation (Voorneveld, 2003).

We denote noop $=_{\mathrm{def}} \langle 0, j, 0\rangle$ as the action of not bidding, for some arbitrary $j\in\mathrm{G}$.

Lemma 3.2. *For each agent* $\mathsf{i}\in\mathrm{N}$*, each bid-tree* $\beta\in\mathcal{L}_{\mathsf{TBBL}}$ *and each* $\boldsymbol{\lambda}\in\mathrm{I}^{\mathsf{nm}}$*,* $v_\mathsf{i}(\mathrm{noop}, \boldsymbol{\lambda}) = 0$*.*

Proof. Remind noop denotes a leaf bid $\langle 0, j, 0\rangle$, where $j\in\mathrm{G}$. Thus, $b_\mathsf{i}(\mathrm{noop}) = 0$. Let $\boldsymbol{\lambda}\in\mathrm{I}^{\mathsf{nm}}$. The value of $\boldsymbol{\lambda}$ given noop, *i.e.*, $v_\mathsf{i}(\mathrm{noop}, \boldsymbol{\lambda})$, is the maximal sum of $b_\mathsf{i}(\beta)\cdot\mathrm{sat}_\mathsf{i}(\beta)$ in a solution sat_i, for all $\beta\in\mathrm{Node}(\mathrm{noop})$. Since $\mathrm{Node}(\mathrm{noop}) = \{\mathrm{noop}\}$ and $b_\mathsf{i}(\mathrm{noop}) = 0$, for any solution sat_i, $v_\mathsf{i}(\mathrm{noop}, \boldsymbol{\lambda}) = 0$. □

Next, we illustrate how to represent protocols with TBBL in ADL.

3.2.2 Vickrey–Clarke–Groves Mechanism

Using TBBL to determinate the action set, let us now represent the Vickrey–Clarke–Groves (VCG) mechanism in ADL. This mechanism chooses the outcome maximizing the reported preferences (Krishna, 2009). Each agent's payment corresponds to the damage she causes the other players, that is, the difference in the social welfare of others with and without her participation (Nisan et al., 2007). We detail the rules specification and the semantic representation. Then, we revisit mechanism properties and evaluate whether the protocol is well-formed.

To represent a VCG mechanism in the combinatorial exchange setting, we first describe its signature, written $\mathcal{S}_{vcg} = (\mathrm{N}, \mathrm{G}, \mathcal{B}, \Phi, \mathrm{Y}, \mathrm{I}, \mathcal{F})$, where $\mathrm{N} = \{1, \ldots, \mathsf{n}\}$, $\mathrm{G} = \{1, \ldots, \mathsf{m}\}$, $\mathcal{B} \subseteq \mathcal{L}_{\mathsf{TBBL}}$, $\Phi = \{\mathrm{bidRound}\}$, $\mathrm{Y} = \{\mathrm{trade}_{\mathsf{i},j}, \mathrm{pay}_{\mathsf{i}} : \mathsf{i} \in \mathrm{N}, j \in \mathrm{G}\}$, and $\mathrm{I} \subset \mathbb{Z}$. Finally, $\mathcal{F}$ contains the functions $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda})$, $\mathrm{WD}_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, X)$ and $\mathrm{WD}_{\lambda_{\mathsf{i},j}}(\boldsymbol{\beta}, X)$ described in the previous section as well as the functions denoting basic mathematical operations (*e.g.*, $\mathrm{sum}(z_1, z_2)$). We also assume $\mathcal{F}$ contains the function $\mathrm{WD}^{-\mathsf{r}}_{\boldsymbol{\lambda}} : \mathcal{B}^{\mathsf{n}} \times \mathrm{I}^{\mathsf{nm}} \to \mathrm{I}^{\mathsf{nm}}$ for any two agents i and r. $\mathrm{WD}^{-\mathsf{r}}_{\boldsymbol{\lambda}}$ is defined exactly like $\mathrm{WD}_{\boldsymbol{\lambda}}$ except that the set N in the winner determination (see Def. 3.2) is replaced by $\mathrm{N} \setminus \{\mathsf{r}\}$ and that the resulting trade for agent r and each good j is equal to zero.

Each instance of a VCG is specific and is defined with respect to $\mathcal{B}$, I and the constant values $\mathsf{n}, \mathsf{m} \in \mathrm{I}_{\succ 0}$ (the size of N and G, resp.), and $\mathsf{X} = (\mathsf{x}_j)_{j \in \mathrm{G}}$, where $\mathsf{x}_{\mathsf{i},j} \in \mathrm{I}_{\succeq 0}$, for each $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$. Each constant $\mathsf{x}_{\mathsf{i},j}$ represents the number of units of j initially held by agent i. The rules of VCG are represented by ADL-formulae as shown in Figure 3.4.

1. $\mathrm{initial} \to \mathrm{bidRound} \wedge \bigwedge_{\mathsf{i} \in \mathrm{N}} (\mathrm{pay}_{\mathsf{i}} = 0 \wedge \bigwedge_{j \in \mathrm{G}} \mathrm{trade}_{\mathsf{i},j} = 0)$
2. $\mathrm{terminal} \leftrightarrow \neg \mathrm{initial}$
3. $\mathrm{terminal} \to \mathrm{legal}_{\mathsf{i}}(\mathrm{noop})$, for each $\mathsf{i} \in \mathrm{N}$
4. $\mathrm{initial} \to \mathrm{legal}_{\mathsf{i}}(\beta)$, for each $\mathsf{i} \in \mathrm{N}$, $\beta \in \mathcal{B}$
5. $\mathrm{does}(\boldsymbol{\beta}) \wedge \mathrm{initial} \to \bigcirc(\bigwedge_{\mathsf{i} \in \mathrm{N}, j \in \mathrm{G}} \mathrm{trade}_{\mathsf{i},j} = \mathrm{WD}_{\lambda_{\mathsf{i},j}}(\boldsymbol{\beta}, \mathsf{X}))$, for each $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$
6. $\mathrm{does}(\beta) \wedge \neg \mathrm{terminal} \wedge \mathrm{p}_{\mathsf{i}} = \mathrm{sub}(\mathrm{sum}_{\mathsf{r} \in \mathrm{N} \setminus \{\mathsf{i}\}}(v_{\mathsf{r}}(\beta_{\mathsf{r}}, \mathrm{WD}^{-\mathsf{i}}_{\boldsymbol{\lambda}}(\beta, \mathsf{X}))),$ $\mathrm{sum}_{\mathsf{r} \in \mathrm{N} \setminus \{\mathsf{i}\}}(v_{\mathsf{r}}(\beta_{\mathsf{r}}, \mathrm{WD}_{\boldsymbol{\lambda}}(\beta, \mathsf{X})))) \to \bigcirc \mathrm{pay}_{\mathsf{i}} = \mathrm{p}_{\mathsf{i}}$, for each $p_{\mathsf{i}} \in \mathrm{I}$, $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$ and $\mathsf{i} \in \mathrm{N}$
7. $\bigcirc \neg \mathrm{bidRound}$

Figure 3.4: Vickrey–Clarke–Groves mechanism represented by Σ_{vcg}

In the initial state, the trade and payment are zero for every agent and good (Rule 1). Any state that is not initial is terminal (Rule 2). The proposition bidRound helps to distinguish the initial state from the terminal state where no trade or payment were assigned to any agent (*e.g*, when all agents bid noop). Once in a terminal state, players can only do noop. Otherwise, they can bid any $\beta \in \mathcal{B}$ (Rules 3 and 4). After performing a joint bid, in the next state each agent receives a trade for each good, which is assigned by the winner determination over the initial allocation and their bids (Rule 5). After a joint bid in the initial state, the payment for agent i will be the difference in the others' welfare with and without her participation (Rule 6). Finally, the proposition bidRound is always false in the next state (Rule 7).

Notice we could as well represent winner determination explicitly in ADL by

capturing the trade maximizing the social welfare among all trades that satisfy constraints C1-C4 (Definition 3.2). For instance, constraint C1 can be written in $\mathcal{L}_{\mathsf{ADL}}$ as $\text{feasible}((\boldsymbol{\lambda_j})_{j\in \mathrm{G}}) =_{\text{def}} \bigwedge_{\mathsf{i}\in \mathrm{N}, j\in \mathrm{G}} \text{sum}(\lambda_{\mathsf{i},j}, x_{\mathsf{i},j}) \geq 0$. We illustrate VCG with an indirect representation of the winner determination for succincteness and clarity of Σ_{vcg}.

3.2.2.1 Representing as a model

Next, we address the model representation. Let $\mathcal{M}_{vcg}$ be the set of ST-models M_{vcg} defined for any $\mathcal{B} \subseteq \mathcal{L}_{\mathsf{TBBL}}$, $\mathrm{I} \subset \mathbb{Z}$, and the constants $\mathsf{n}, \mathsf{m} \in \mathrm{I}_{\succ 0}$ and $\mathsf{X} = (\mathsf{x_j})_{j\in \mathrm{G}}$, where $\mathsf{x}_{\mathsf{i},j} \in \mathrm{I}_{\succeq 0}$, for each $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$. Each M_{vcg} is defined as follows:

- $\mathrm{W} = \{\langle b, (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}, \mathrm{p}\rangle : b \in \{0, 1\} \ \&\ \mathrm{p}_{\mathsf{i}}, \lambda_{\mathsf{i},j} \in \mathrm{I} \ \&\ \mathsf{i} \in \mathrm{N} \ \&\ j \in \mathrm{G}\}$;
- $\bar{\mathrm{w}} = \langle 1, 0, \ldots, 0, 0, \ldots, 0\rangle$;
- $\mathrm{T} = \mathrm{W} \setminus \{\bar{\mathrm{w}}\}$;
- $\mathrm{L} = \{(\mathrm{w}, \mathsf{i}, \text{noop}) : \mathsf{i} \in \mathrm{N} \ \&\ \mathrm{w} \in \mathrm{T}\} \cup \{(\bar{\mathrm{w}}, \mathsf{i}, \beta) : \beta \in \mathcal{B} \ \&\ \mathsf{i} \in \mathrm{N}\}$;
- U is defined as follows: for all $\mathrm{w} = \langle b, (\boldsymbol{\lambda_j})_{j\in \mathrm{G}}, \mathrm{p}\rangle \in \mathrm{W}$ and for all $\boldsymbol{d} \in \mathcal{B}^{\mathsf{m}}$:
  - If $\mathrm{w} = \bar{\mathrm{w}}$, then $\mathrm{U}(\mathrm{w}, \boldsymbol{d}) = \langle 0, (\boldsymbol{\lambda'_j})_{j\in \mathrm{G}}, \mathrm{p'}\rangle\rangle$, where each component is updated as follows, for each $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$. The number of units j traded for agent i is given by the winner determination: $\lambda'_{\mathsf{i},j} = \mathrm{WD}_{\lambda_{\mathsf{i},j}}(\boldsymbol{d}, \mathsf{X})$. The payment for i is the difference between the social welfare of others with and without i's participation:

$$\mathrm{p}'_{\mathsf{i}} = \sum_{\mathsf{r}\in \mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(d_{\mathsf{r}}, \mathrm{WD}^{-\mathsf{i}}_{\boldsymbol{\lambda}}(\boldsymbol{d}, \mathsf{X})) - \sum_{\mathsf{r}\in \mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(d_{\mathsf{r}}, \mathrm{WD}_{\boldsymbol{\lambda}}(\boldsymbol{d}, \mathsf{X}))$$

  - Otherwise, $\mathrm{U}(\mathrm{w}, \boldsymbol{d}) = \mathrm{w}$.
- For each $\mathrm{w} \in \mathrm{W}$, $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$, $\pi_{\mathrm{Y}}(\mathrm{w}, \text{trade}_{\mathsf{i},j}) = \lambda_{\mathsf{i},j}$; $\pi_{\mathrm{Y}}(\mathrm{w}, \text{pay}_{\mathsf{i}}) = \mathrm{p}_{\mathsf{i}}$; and $\pi_{\Phi}(\mathrm{w}) = \{\text{bidRound} : b = 1\}$.

Hereafter, we assume an instance of $\mathrm{M}_{vcg} \in \mathcal{M}_{vcg}$ and Σ_{vcg} for some $\mathcal{B} \subseteq \mathcal{L}_{\mathsf{TBBL}}$, $\mathrm{I} \subset \mathbb{Z}$, $\mathsf{n}, \mathsf{m} \in \mathrm{I}_{\succ 0}$ and $\mathsf{x}_{\mathsf{i},j} \in \mathrm{I}_{\succeq 0}$, where $\mathsf{i} \in \mathrm{N}$, $j \in \mathrm{G}$.

Example 3.2. Let $\mathrm{M}_{vcg} \in \mathcal{M}_{vcg}$ such that the sets of agents and goods are the same from Example 3.1 and the initial allocation is as follows: $\mathsf{x}_{\mathsf{i},\mathsf{a}} = 0$, $\mathsf{x}_{\mathsf{i},\mathsf{b}} = 1$, $\mathsf{x}_{\mathsf{s},\mathsf{a}} = 2$ and $\mathsf{x}_{\mathsf{s},\mathsf{b}} = 0$ (*i.e.*, at the beginning of the auction, agent i has 1 unit of b and agent s has 2 units of a). Figure 3.5 illustrates a path in M_{vcg}, where the agents perform the bids previously introduced in Figure 3.3. In state w_0, all the payments and trades are zero. Their joint bid leads to state w_1, where the trade obtained by the winner determination is $(2, -1, -2, 1)$. The tie-breaking ensures that the trade is unique. Thus, in w_1 agent i has 2 units of a and agent s has 1 unit of b. Since w_1 is terminal, the agents can only bid noop.

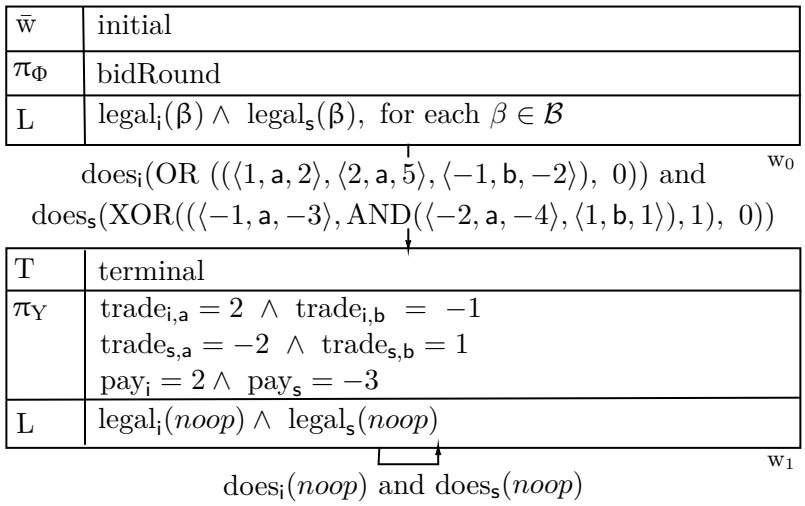

Figure 3.5: A Path in M_{vcg}, with 2 bidders and 2 goods

3.2.2.2 Evaluating the protocol

As for SAA, let us now evaluate the protocol representing a VCG mechanism. First, Lemma 3.3 shows that M_{vcg} is a sound representation of Σ_{vcg}.

Lemma 3.3. *M_{vcg} is an ST-model and it is a model of Σ_{vcg}.*

Proof. (Sketch) It is routine to check that M_{vcg} is actually an ST-model. Given a path δ in M_{vcg} and a stage t of δ, we need to show that $M_{vcg}, \delta, t \models \varphi$, for each $\varphi \in \Sigma_{vcg}$.

Let us verify Rule 1. Assume $M_{vcg}, \delta, t \models \text{initial}$, then $\delta[t] = \bar{w}$, *i.e.*, $\delta[t] = \langle 1, 0, \ldots, 0, 0, \ldots, 0\rangle$. By the definitions of π_Y and π_Φ, $\pi_\Phi(\bar{w}) = \{\text{bidRound}\}$, $\pi_Y(\bar{w}, \text{pay}_i) = 0$ and $\pi_Y(\bar{w}, \text{trade}_{i,j}) = 0$ for all $i \in N$ and $j \in G$. Thus, $M_{vcg}, \delta, t \models \text{bidRound} \wedge \bigwedge_{i \in N} \text{pay}_i = 0 \wedge \bigwedge_{j \in G} \text{trade}_{i,j} = 0$.

Now we verify Rule 4. Assume $M_{vcg}, \delta, t \models \text{initial}$, then $\delta[t] = \bar{w}$ and for all $i \in N$ and $\beta \in \mathcal{B}$, $(\bar{w}, i, \beta) \in L$. Thus, $M_{vcg}, \delta, t \models \text{legal}_i(\beta)$.

Then we consider Rule 5. $M_{vcg}, \delta, t \models \text{does}(\boldsymbol{\beta}) \wedge \text{initial}$, for $\boldsymbol{\beta} \in \mathcal{B}^m$, *i.e.*, $M_{vcg}, \delta, t \models \bigwedge_{i \in N} \text{does}(\beta_i)$ and $M_{vcg}, \delta, t \models \text{initial}$. Thus, $\theta_i(\delta, t) = \beta_i$ for all $i \in N$. The update function U defines $\delta[t+1]$ such that $\pi_Y(\delta[t+1], \text{trade}_{i,j}) = \text{WD}_{\lambda_{i,j}}(\boldsymbol{\beta}, X)$, for each $i \in N$ and $j \in G$. Thus, $M_{vcg}, \delta, t+1 \models \bigwedge_{i \in N, j \in G} \text{trade}_{i,j} = \text{WD}_{\lambda_{i,j}}(\boldsymbol{\beta}, X)$ and also $M_{vcg}, \delta, t \models \bigcirc(\bigwedge_{i \in N, j \in G} \text{trade}_{i,j} = \text{WD}_{\lambda_{i,j}}(\boldsymbol{\beta}, X))$. Using the abbreviation, $M_{vcg}, \delta, t \models \bigcirc(\bigwedge_{i \in N} \text{trade}_i = \text{WD}_{\lambda_i}(\boldsymbol{\beta}, X))$.

The remaining rules are verified in a similar way. □

Next, we show Σ_{vcg} is a well-formed protocol, that is, each path in M_{vcg} reaches a terminal state and there is a legal action for each agent in all reachable states.

Theorem 3.3. *Σ_{vcg} is a well-formed protocol over the ST-model M_{vcg}.*

Proof. Since M_{vcg} is a model of Σ_{vcg} (see Lemma 3.3), we have to show that for each path δ in M_{vcg} and each agent $\mathsf{i} \in \mathrm{N}$,

1. δ is a complete path;
2. $\mathrm{M}_{vcg} \models \bigvee_{a \in \mathcal{B}} \mathrm{legal}_{\mathsf{i}}(a)$.

Given a path δ in M_{vcg} and a stage t of δ. Let us verify Statement 1. We show that $\mathrm{M}_{vcg} \models \mathrm{initial} \rightarrow \bigcirc\mathrm{terminal}$. Assume $\mathrm{M}_{vcg}, \delta, t \models \mathrm{initial}$. Then, $\delta[t] = \bar{\mathrm{w}}$. By the path definition, for any $j \geq 1$, $\delta[j] \neq \bar{\mathrm{w}}$. By the construction of T, we have $\mathrm{T} = \mathrm{W} \setminus \{\bar{\mathrm{w}}\}$. Thus, $\mathrm{M}_{vcg}, \delta, t+1 \models \mathrm{terminal}$ and $\mathrm{M}_{vcg}, \delta, t \models \bigcirc\mathrm{terminal}$.

Statement 2 is straightforward from Rules 3 and 4 from Σ_{vcg}. □

The next lemma shows that if an agent bids noop in an initial state, her payment will be zero. Furthermore, if the payment is zero in a terminal state, it will be zero in the succeeding state.

Lemma 3.4. *For each agent* $\mathsf{i} \in \mathrm{N}$, $\mathrm{M}_{vcg} \models \mathrm{initial} \wedge \mathrm{does}_{\mathsf{i}}(\mathrm{noop}) \rightarrow \bigcirc \mathrm{pay}_{\mathsf{i}} = 0 \wedge \bigwedge_{j \in \mathrm{G}} \mathrm{trade}_{\mathsf{i},j} \geq 0$

Proof. Straightforward from Lemma 3.2, Rule 6 from Σ_{vcg} and Rule R2 from the definition of b_{i}. □

We then focus on properties from Mechanism Design, that is budget balance, individual rationality, efficiency and strategyproofness. These results for VCG have already been proved (Krishna, 2009; Nisan et al., 2007) and here we show how they are rephrased and verified with ADL. First, due to the VCG payments, M_{vcg} is not budget balanced.

Proposition 3.5. $\mathrm{M}_{vcg} \not\models \bigcirc\mathrm{SBB}$ *and* $\mathrm{M}_{vcg} \not\models \bigcirc\mathrm{WBB}$.

Proof. We show $\mathrm{M}_{vcg} \not\models \bigcirc\mathrm{WBB}$ and $\mathrm{M}_{vcg} \not\models \bigcirc\mathrm{SBB}$ by showing a counter example. Given two distinct agents i, r in N and a good j in G, let δ be a path in M_{vcg} such that $\theta_{\mathsf{i}}(\delta, 0) = \langle 1, j, 5 \rangle$ and $\theta_{\mathsf{r}}(\delta, 0) = \langle -1, j, -3 \rangle$. For any other agent $\mathsf{s} \in \mathrm{N} \setminus \{\mathsf{i}, \mathsf{r}\}$, $\theta_{\mathsf{s}}(\delta, 0) = \mathit{noop}$. Since this actions are legal in the initial state $\delta[0]$, there exists such path.

By Rule 5 and the definition of function $\mathrm{WD}_{\boldsymbol{\lambda}}$, we have that $\mathrm{M}_{vcg}, \delta, 1 \models (\mathrm{trade}_{\mathsf{i},j} = 1 \wedge \mathrm{trade}_{\mathsf{r},j} = -1)$ and $\mathrm{M}_{vcg}, \delta, 1 \models \mathrm{trade}_{\mathsf{s},j'} = 0$ for all pairs $(\mathsf{s}, j') \neq (\mathsf{i}, j)$ and $(\mathsf{s}, j') \neq (\mathsf{r}, j)$. That is, the good j is sold by r and bought by i. The social welfare of all agents other than i is -3 and the social welfare of all agents other than r is 5. If either i or r did not participate, there would be no trade and the social welfare would be zero. Thus, by Rule 6, the payments for i and r are 3 and -5, resp., that is, $\mathrm{M}_{vcg}, \delta, 1 \models \mathrm{pay}_{\mathsf{i}} = 3 \wedge \mathrm{pay}_{\mathsf{r}} = -5$. The other agents' do not have payments on $\delta[1]$, *i.e.*, $\mathrm{M}_{vcg}, \delta, 1 \models \bigwedge_{\mathsf{s} \in \mathrm{N} \setminus \{\mathsf{i},\mathsf{r}\}} \mathrm{pay}_{\mathsf{s}} = 0$. Then, $\mathrm{M}_{vcg}, \delta, 1 \models \mathrm{sum}_{\mathsf{i} \in \mathrm{N}}(\mathrm{pay}_{\mathsf{i}}) < 0$. Thus we have that $\mathrm{M}_{vcg}, \delta, 0 \not\models \bigcirc\mathrm{SBB}$ and $\mathrm{M}_{vcg}, \delta, 0 \not\models \bigcirc\mathrm{WBB}$. □

After the agents report their preferences, M_{vcg} chooses the trade maximizing the social welfare (*i.e.*, the cumulative of values for the trade given the agents' bids).

Proposition 3.6. *Given a joint action* $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$, $\mathrm{M}_{vcg} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{EF}(\boldsymbol{\beta})$

Proof. (Sketch) Since $\mathrm{M}_{vcg} \models \mathrm{initial} \rightarrow \bigcirc\mathrm{terminal}$ and $\delta[t+1] = \delta[t]$ whenever $\delta[t] \in \mathrm{T}$, it suffices to show that $\mathrm{M}_{vcg}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{EF}(\boldsymbol{\beta})$, for any joint action $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$. Assume $\mathrm{M}_{vcg}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta})$, by Rule 5 we have that $\mathrm{M}_{vcg}, \delta, 1 \models \bigwedge_{\mathsf{i}\in\mathrm{N}, j\in\mathrm{G}} \mathrm{trade}_{\mathsf{i},j} = \mathrm{WD}_{\lambda_{\mathsf{i},j}}(\boldsymbol{\beta}, \mathsf{X})$. By the definition of the winner determination function $\mathrm{WD}_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, \mathsf{X})$ (and consequently function $\mathrm{WD}_{\lambda_{\mathsf{i},j}}$ for each agent i and good j), we have that the trade $(\pi_{\mathrm{Y}}(\delta[1], \mathrm{trade}_{\mathsf{i},j}))_{j\in\mathrm{G},\mathsf{i}\in\mathrm{N}}$ maximizes the cumulative value among the agents. That is, $\mathrm{M}_{vcg}, \delta, 1 \models \mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_{\boldsymbol{j}})_{j\in\mathrm{G}})) = \max_{\boldsymbol{\lambda}\in\Lambda}(\mathrm{sum}_{\mathsf{i}\in\mathrm{N}}(v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda})))$ and $\mathrm{M}_{vcg}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{EF}(\boldsymbol{\beta})$. □

The VCG mechanism is individually rational when the agents' preferences over trades are non-negative (Nisan et al., 2007).

Proposition 3.7. *Given a joint action* $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$, *if* $\vartheta_{\mathsf{i}}(\boldsymbol{\lambda}) \geq 0$ *for all* $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$, $\vartheta_{\mathsf{i}} \in V_{\mathsf{i}}$ *and* $\mathsf{i} \in \mathrm{N}$, *then* $\mathrm{M}_{vcg} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{IR}(\boldsymbol{\beta})$.

Proof. Let $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$ be a joint action and δ be a path in M_{vcg}. Since $\mathrm{M}_{vcg} \models \mathrm{initial} \rightarrow \bigcirc\mathrm{terminal}$ and $\delta[t+1] = \delta[t]$ whenever $\delta[t] \in \mathrm{T}$, it suffices to show that $\mathrm{M}_{vcg}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc\mathrm{IR}(\boldsymbol{\beta})$. Assume $\mathrm{M}_{vcg}, \delta, 0 \models \mathrm{does}(\boldsymbol{\beta})$ and that the preferences represented by $\mathrm{L}(\delta[1], \mathsf{i})$ are non-negative for every agent and possible trade. That is, $v_{\mathsf{i}}(\beta, \boldsymbol{\lambda}) \geq 0$ for any bid β, trade $\boldsymbol{\lambda}$ and agent i.

Let $\boldsymbol{\lambda} = (\boldsymbol{\lambda_j})_{j\in\mathrm{G}}$ denote the trade performed after the agents report $\boldsymbol{\beta}$, where $\lambda_{\mathsf{i},j} = \pi_{\mathrm{Y}}(\delta[1], \mathrm{trade}_{\mathsf{i},j}) = \mathrm{WD}_{\lambda_{\mathsf{i},j}}(\boldsymbol{\beta}, \mathsf{X})$ for each good j and agent i. We denote by $\boldsymbol{\lambda}^{-\mathsf{i}} = \mathrm{WD}^{-\mathsf{i}}_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, \mathsf{X})$ the trade that would happen if i did not participate.

The utility of agent i in $\delta[1]$ is $u_{\mathsf{i}} = v(\beta_{\mathsf{i}}, \boldsymbol{\lambda}) - \pi_{\mathrm{Y}}(\delta[1], \mathrm{pay}_{\mathsf{i}})$. According to the payment definition (Rule 6), agent i's utility in $\delta[1]$ is

$$u_{\mathsf{i}} = v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda}) - \Big(\sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}}) - \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda})\Big)$$

or

$$u_{\mathsf{i}} = v_{\mathsf{i}}(\beta_{\mathsf{i}}, \boldsymbol{\lambda}) - \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}}) + \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda})$$

or simply

$$u_{\mathsf{i}} = \sum_{\mathsf{r}\in\mathrm{N}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}) - \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}})$$

Thus, agent i's utility is non-negative if

$$\sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}) \geq \sum_{\mathsf{r}\in\mathrm{N}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}})$$

Assume for contradiction that this is not the case, that is,

$$\sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}}) > \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda})$$

Since $v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}}) \geq 0$, then we have

$$\sum_{\mathsf{r}\in\mathrm{N}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}}) \geq \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}}) > \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda})$$

which is a contradiction since $\boldsymbol{\lambda}$ is the efficient trade (see Proposition 3.6). Thus, $\mathrm{M}_{vcg}, \delta, 1 \models \bigwedge_{\mathsf{r}\in\mathrm{N}} \mathrm{sub}(v_{\mathsf{r}}(\beta_{\mathsf{r}}, (\mathrm{trade}_j)_{j\in\mathrm{G}}), \mathrm{pay}_{\mathsf{r}}) \geq 0$ and $\mathrm{M}_{vcg}, \delta, 0 \models \bigcirc\mathrm{IR}(\boldsymbol{\beta})$. □

The VCG mechanism is also strategyproof (Nisan et al., 2007), because the bid of an agent does not influence her payment.

Theorem 3.4. M_{vcg} *is strategyproof.*

Proof. Given a path δ in M_{vcg} and a stage $t \geq 0$ in δ, such that $\mathrm{L}(\delta[t], \mathsf{i}) \approx_{\mathsf{i}} V_{\mathsf{i}}$. Since $\mathrm{M}_{vcg} \models \text{initial} \rightarrow \bigcirc\text{terminal}$ and $\delta[t+1] = \delta[t]$ whenever $\delta[t] \in \mathrm{T}$, it suffices to show consider the case where $t = 0$.

Let $\boldsymbol{\vartheta} \in \prod_{\mathsf{i}\in\mathrm{N}} V_{\mathsf{i}}$ be a preference profile and $\delta_{\boldsymbol{\vartheta}}$ denote a path such that $\delta[0] = \delta_{\boldsymbol{\vartheta}}[0]$ and $\theta(\delta_{\boldsymbol{\vartheta}}, 0) = \boldsymbol{\beta}_{\boldsymbol{\vartheta}}$. We let $\vartheta'_{\mathsf{i}} \in V_{\mathsf{i}}$ denote a preference of agent i, $\boldsymbol{\vartheta'} = (\vartheta'_{\mathsf{i}}, \vartheta_{-\mathsf{i}})$ and $\delta_{\boldsymbol{\vartheta'}}$ be a path such that $\delta[0] = \delta_{\boldsymbol{\vartheta'}}[0]$, $\theta_{\mathsf{i}}(\delta_{\boldsymbol{\vartheta'}}, 0) = \beta_{\vartheta'_{\mathsf{i}}}$ and $\theta_{\mathsf{r}}(\delta_{\boldsymbol{\vartheta'}}, 0) = \beta_{\vartheta_{\mathsf{r}}}$ for each agent $\mathsf{r} \neq \mathsf{i}$. That is, $\mathrm{M}_{vcg}, \delta_{\boldsymbol{\vartheta}}, 0 \models \mathrm{does}(\boldsymbol{\beta}_{\boldsymbol{\vartheta}})$ and $\mathrm{M}_{vcg}, \delta_{\boldsymbol{\vartheta'}}, 0 \models \mathrm{does}_{\mathsf{i}}(\beta_{\vartheta'_{\mathsf{r}}}) \wedge \bigwedge_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} \mathrm{does}_{\mathsf{r}}(\beta_{\vartheta_{\mathsf{r}}})$.

Let $u_{\vartheta_{\mathsf{i}}} \in \mathrm{I}$ such that $\mathrm{M}_{vcg}, \delta_{\boldsymbol{\vartheta}}, 0 \models \bigcirc\mathrm{sub}(v_{\mathsf{i}}(\beta_{\vartheta_{\mathsf{i}}}, (\mathrm{trade}_j)_{j\in\mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) = u_{\vartheta_{\mathsf{i}}}$ holds. As we saw in the proof of Proposition 3.7, agent i's utility in $\delta_{\boldsymbol{\vartheta}}[1]$ is simply

$$u_{\vartheta_{\mathsf{i}}} = \sum_{\mathsf{r}\in\mathrm{N}} v_{\mathsf{r}}(\beta_{\vartheta_{\mathsf{r}}}, \boldsymbol{\lambda}) - \sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\vartheta_{\mathsf{r}}}, \boldsymbol{\lambda}^{-\mathsf{i}})$$

where $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_j)_{j\in\mathrm{G}}$ denote the trade performed in $\delta_{\boldsymbol{\vartheta}}[1]$ with $\lambda_{\mathsf{i},j} = \pi_{\mathrm{Y}}(\delta_{\boldsymbol{\vartheta}}[1], \mathrm{trade}) = \mathrm{WD}_{\lambda_{\mathsf{i},j}}(\boldsymbol{\beta}, \mathsf{X})$ for each good j and agent i, and $\boldsymbol{\lambda}^{-\mathsf{i}} = \mathrm{WD}^{-\mathsf{i}}_{\boldsymbol{\lambda}}(\boldsymbol{\beta}, \mathsf{X})$ is the trade that would happen if i did not participate in the auction.

Notice that the bid of i has no impact in $\sum_{\mathsf{r}\in\mathrm{N}\setminus\{\mathsf{i}\}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda}^{-\mathsf{i}})$ by the definition of $\mathrm{WD}^{-\mathsf{i}}_{\boldsymbol{\lambda}}$. Thus, it means that the bid maximizing i's utility is the one that maximize

$$\sum_{\mathsf{r}\in\mathrm{N}} v_{\mathsf{r}}(\beta_{\mathsf{r}}, \boldsymbol{\lambda})$$

which, by definition, is the case when she bids truthfully. That is, $u_{\vartheta_{\mathsf{i}}}$ is the maximum utility i obtains in the succeeding stages of all paths starting in $\delta[0]$. Thus, $\mathrm{M}_{vcg}, \delta_{\boldsymbol{\vartheta}}, 0 \models \bigcirc\mathrm{sub}(v_{\mathsf{i}}(\beta_{\vartheta_{\mathsf{i}}}, (\mathrm{trade}_j)_{j\in\mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) = u_{\vartheta_{\mathsf{i}}}$ and for any ϑ', $\mathrm{M}_{vcg}, \delta_{\boldsymbol{\vartheta'}}, 0 \models \bigcirc\mathrm{sub}(v_{\mathsf{i}}(\beta_{\vartheta_{\mathsf{i}}}, (\mathrm{trade}_j)_{j\in\mathrm{G}}), \mathrm{pay}_{\mathsf{i}}) \leq u_{\vartheta_{\mathsf{i}}}$. □

We conclude the section by discussing variants of combinatorial exchange. In combinatorial auctions, there are two types of participants: buyers and sellers. The main difference to a combinatorial exchange is that buyers can only demand for non-negative quantities and prices while sellers can only ask for non-positive quantities and prices. These restrictions can be easily encoded in ADL by including a proposition for denoting the participants' types and defining legality rules based on their types. In Section 3.1, we exemplify the representation of a single-sided auction. Restricting the number of participants with the type buyer (or similarly

seller) to one is an alternative way of encoding single-sided auctions. Multi-unit auctions with single items can be represented for different market structures (*e.g.*, the single-sided setting) by restricting the number of good types to one (that is, $|\mathrm{G}| = 1$). In the next section, we show how to represent an iterative protocol in ADL using a slightly different version of TBBL.

3.2.3 Iterative Combinatorial Exchange

We conclude the section on combinatorial exchange by considering an iterative protocol, denoted ICE. The protocol is a simplified and first-price version of the mechanism presented in (Parkes et al., 2005). In this auction, bidders report an interval of prices they are willing to pay (or receive) for a trade. For ensuring termination, agents need to refine their bids, that is, to shrink the value interval reported in their previous bid. For expressing such conditions, we consider the TBBL extension with value bounds, denoted $\mathcal{L}_{\mathsf{TBBL}+}$. In this variant proposed by Lubin et. al (Lubin et al., 2008), agents report a pair of valuation bounds in each node of their bid. The syntax of $\mathcal{L}_{\mathsf{TBBL}+}$ is obtained by replacing the value v of a leaf node $\langle \mathsf{q}, j, \mathsf{v} \rangle$ by the lower bound $\underline{\mathsf{v}} \in \mathrm{I}$ and upper bound $\overline{\mathsf{v}} \in \mathrm{I}$ with $\underline{\mathsf{v}} \leq \overline{\mathsf{v}}$. The bounds $\underline{\mathsf{v}}$ and $\overline{\mathsf{v}}$ denote the minimum and maximum price the bidder considers acceptable to pay (or receive) for q units of j, respectively. Bid nodes with IC operators are similarly updated to include value bounds.

Given an agent $\mathsf{i} \in \mathrm{N}$, a bid $\beta \in \mathcal{L}_{\mathsf{TBBL}+}$ and a trade $\lambda \in \mathrm{I}^{\mathsf{nm}}$, the functions $\underline{v}_{\mathsf{i}}(\beta, \lambda)$, $\overline{v}_{\mathsf{i}}(\beta, \lambda)$, $\underline{b}_{\mathsf{i}}(\beta)$, $\overline{b}_{\mathsf{i}}(\beta)$ are defined with the same semantics from the functions v_{i} and b_{i} introduced in Section 3.2. Since the lower bound is no greater than the upper bound, we have that $\underline{b}_{\mathsf{i}}(\beta) \leq \overline{b}_{\mathsf{i}}(\beta)$ and $\underline{v}_{\mathsf{i}}(\beta, \lambda) \leq \overline{v}_{\mathsf{i}}(\beta, \lambda)$. For representing this ICE protocol with ADL, we first fix a constant $\varepsilon \in [0, 1]$ for estimating the weight of the bounds in the value of a bid given a trade (*i.e.*, function v_{i}). If $\varepsilon = 1$, the bid value is based only on its lower bounds. Likewise, the upper bounds determinate the bid value when $\varepsilon = 0$. Next, we describe the auction signature, written $\mathcal{S}_{ice} = (\mathrm{N}, \mathrm{G}, \mathcal{B}, \{\}, \mathrm{Y}, \mathrm{I}, \mathcal{F})$, where $\mathrm{N} = \{1, \ldots, \mathsf{n}\}$, $\mathrm{G} = \{1, \ldots, \mathsf{m}\}$, $\mathcal{B} \subseteq \mathcal{L}_{\mathsf{TBBL}+}$, $\mathrm{Y} = \{\mathrm{trade}_{\mathsf{i},j}, \mathrm{pay}_{\mathsf{i}} : \mathsf{i} \in \mathrm{N}, j \in \mathrm{G}\}$, and $\mathrm{I} \subset \mathbb{Z}$. If $\langle \mathsf{q}, j, \underline{\mathsf{v}}, \overline{\mathsf{v}} \rangle$ is a bid in $\mathcal{B}$, we assume $\langle \mathsf{q}, j, \underline{\mathsf{v}}', \overline{\mathsf{v}}' \rangle \in \mathcal{B}$ for any $\underline{\mathsf{v}}' \geq \underline{\mathsf{v}}$ and $\overline{\mathsf{v}}' \leq \overline{\mathsf{v}}$ such that $\underline{\mathsf{v}}' \leq \overline{\mathsf{v}}'$. Similarly, if $\mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, \underline{\mathsf{v}}, \overline{\mathsf{v}}) \in \mathcal{B}$, we assume $\mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, \underline{\mathsf{v}}', \overline{\mathsf{v}}') \in \mathcal{B}$ for any $\underline{\mathsf{v}}' \geq \underline{\mathsf{v}}$ and $\overline{\mathsf{v}}' \leq \overline{\mathsf{v}}$ such that $\underline{\mathsf{v}}' \leq \overline{\mathsf{v}}'$. $\mathcal{F}$ contains the basic mathematical operations as well as the following functions: $v_{\mathsf{i}} : \mathcal{B} \times \mathrm{I}^{\mathsf{nm}} \to \mathrm{I}$, $\mathrm{WD}^{\varepsilon}_{\lambda_{\mathsf{i},j}} : \mathcal{B}^{\mathsf{n}} \times \mathrm{I}^{\mathsf{nm}} \to \mathrm{I}$, $\mathrm{eq} : \mathcal{B} \times \mathcal{B} \to [0, 1]$ and $\mathrm{uncert} : \mathcal{B} \to \mathrm{I}$. We next describe each of those functions.

Given an agent i, the value of bid $\beta \in \mathcal{L}_{\mathsf{TBBL}+}$ given a trade $\boldsymbol{\lambda} \in \mathrm{I}^{\mathsf{nm}}$ is defined as follows:

$$v_{\mathsf{i}}(\beta, \boldsymbol{\lambda}) = \lceil \varepsilon \cdot \underline{v}_{\mathsf{i}}(\beta, \boldsymbol{\lambda}) \rceil + \lfloor (1 - \varepsilon) \cdot \overline{v}_{\mathsf{i}}(\beta, \boldsymbol{\lambda}) \rfloor$$

Notice the rounding of the terms[1] ensure the result will not be smaller than the lower bound neither greater than the upper bound.

[1] We denote by $\lfloor x \rfloor$ the greatest integer less than or equal to $x \in \mathbb{R}$ and $\lceil x \rceil$ the least integer greater than or equal to $x \in \mathbb{R}$.

Assuming a joint bid $\boldsymbol{\beta} \in \mathcal{L}^{\mathsf{n}}_{\mathsf{TBBL}+}$ and an initial allocation $X \in \mathrm{I}^{\mathsf{nm}}_{\succeq 0}$, function $\mathrm{WD}^{\varepsilon}_{\lambda_{\mathsf{i}},j}(\boldsymbol{\beta}, X)$ is defined exactly as function $\mathrm{WD}_{\lambda_{\mathsf{i}},j}(\boldsymbol{\beta}, X)$ defined in Section 3.2.1.2, except that the winner determination is replaced by the following:

$$\mathrm{WD}^{\varepsilon}(\boldsymbol{\beta}, X) : \arg\max_{\boldsymbol{\lambda},\mathrm{sat}} \sum_{\mathsf{i}\in \mathrm{N}} \sum_{\beta \in \mathrm{Node}(\beta_{\mathsf{i}})} (\lceil \varepsilon \underline{b}_{\mathsf{i}}(\beta) \cdot \mathrm{sat}_{\mathsf{i}}(\beta) \rceil + \left\lfloor (1-\varepsilon)\overline{b}_{\mathsf{i}}(\beta) \cdot \mathrm{sat}_{\mathsf{i}}(\beta) \right\rfloor)$$

$$\text{s.t. Constraints } C1 - C4 \text{ hold (see Def. 3.2)}$$

Given two bids $\beta, \gamma \in \mathcal{L}_{\mathsf{TBBL}+}$, $\mathrm{eq}(\beta, \gamma)$ denotes whether they are equivalent on structure, in the sense of differing only on their valuation bounds, and it is defined as follows:

- If β is in the form $\langle \mathsf{q}, j, \underline{\mathsf{v}}, \overline{\mathsf{v}} \rangle$ and $\gamma = \langle \mathsf{q}, j, \underline{\mathsf{v}}', \overline{\mathsf{v}}' \rangle$ for some $\underline{\mathsf{v}}', \overline{\mathsf{v}}' \in \mathrm{I}$, then $\mathrm{eq}(\beta, \gamma) = 1$;
- If β is in the form $\mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, \underline{\mathsf{v}}, \overline{\mathsf{v}})$, $\gamma = \mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}', \underline{\mathsf{v}}', \overline{\mathsf{v}}')$ for some $\underline{\mathsf{v}}', \overline{\mathsf{v}}' \in \mathrm{I}$ and each $\bar{\beta}_k \backsim \bar{\beta}'_k$, for each $0 < k_i \leq |\bar{\beta}|$, then $\mathrm{eq}(\beta, \gamma) = 1$;
- Otherwise, $\mathrm{eq}(\beta, \gamma) = 0$.

The difference among the bounds of a node represents its uncertainty. That is, the higher the bound difference, the less precise the node is about the agents' preference. The actual willingness-to-pay (or receive), is unknown except when the lower and upper bounds are the same (Parkes et al., 2005). We define function $\mathrm{uncert}(\langle \mathsf{q}, j, \underline{\mathsf{v}}, \overline{\mathsf{v}} \rangle) = \overline{\mathsf{v}} - \underline{\mathsf{v}}$ and $\mathrm{uncert}(\mathrm{IC}^{\mathrm{x}}_{\mathrm{y}}(\bar{\beta}, \underline{\mathsf{v}}, \overline{\mathsf{v}})) = \overline{\mathsf{v}} - \underline{\mathsf{v}} + \sum_{0<k\leq|\bar{\beta}|} \mathrm{uncert}(\bar{\beta}_k)$ for capturing the uncertainty of bid in $\mathcal{L}_{\mathsf{TBBL}+}$.

Each instance of ICE is specific and is defined with respect to $\mathcal{B}$, I and the constant values $\varepsilon \in [0, 1]$, $\mathsf{n}, \mathsf{m} \in \mathrm{I}_{\succ 0}$ (the size of N and G, resp.), and $\mathsf{X} = (\mathbf{x_j})_{j\in \mathrm{G}}$, where $\mathsf{x}_{\mathsf{i},j} \in \mathrm{I}_{\succeq 0}$, for each $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$. The rules of a ICE are represented by ADL-formulae as shown in Figure 3.6.

In the initial state, there is no payment or trade for any agent (Rule 1) and agents can report any bid (Rule 2). After performing a bid, an agent is allowed to report any bid that has the same structure and less uncertainty than her last bid. When the bid has no uncertainty (*i.e.*, for each node, its lower and upper bounds are the same), the agent must repeat her bid in the next turn (Rules 3 and 4). When there is no uncertainty in all bids performed in a state, the next state is terminal (Rule 5). The agents pay their reported values according to their trade in a given state (Rule 6). Finally, the agents' trades are computed in each round using the winner determination given their bids and their initial allocation (Rule 7).

3.2.3.1 Representing as a model

Next, we address the model representation. Let $\mathcal{M}_{ice}$ be the set of ST-models M_{ice} defined for any $\mathcal{B} \subseteq \mathcal{L}_{\mathsf{TBBL}+}$, $\mathrm{I} \subset \mathbb{Z}$, and the constants $\mathsf{n}, \mathsf{m} \in \mathrm{I}_{\succ 0}$ and $\mathsf{X} = (\mathbf{x_j})_{j\in \mathrm{G}}$, where $\mathsf{x}_{\mathsf{i},j} \in \mathrm{I}_{\succeq 0}$, for each $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$. Each M_{ice} is defined as follows:

1. $\text{initial} \rightarrow \bigwedge_{\mathsf{i}\in\mathrm{N}}(\text{pay}_\mathsf{i} = 0 \wedge \bigwedge_{j\in\mathrm{G}} \text{trade}_{\mathsf{i},j} = 0)$
2. $\text{initial} \rightarrow \text{legal}_\mathsf{i}(\beta)$, for each $\mathsf{i} \in \mathrm{N}$, $\beta \in \mathcal{B}$
3. $\text{does}_\mathsf{i}(\beta) \wedge \text{eq}(\beta, \gamma) \wedge \text{uncert}(\gamma) < \text{uncert}(\beta) \rightarrow \bigcirc\text{legal}_\mathsf{i}(\gamma)$, for each $\mathsf{i} \in \mathrm{N}$, $\beta, \gamma \in \mathcal{B}$
4. $\text{does}_\mathsf{i}(\beta) \wedge \text{uncert}(\beta) = 0 \rightarrow \bigcirc\text{legal}_\mathsf{i}(\beta)$, for each $\mathsf{i} \in \mathrm{N}$ and $\beta \in \mathcal{B}$
5. $(\bigwedge_{\mathsf{i}\in\mathrm{N}} \text{does}_\mathsf{i}(\beta_\mathsf{i}) \wedge \text{uncert}(\beta_\mathsf{i}) = 0) \rightarrow \bigcirc\text{terminal}$, for each $\boldsymbol{\beta} \in \mathcal{B}^\mathsf{n}$
6. $\text{does}_\mathsf{i}(\beta) \rightarrow \bigcirc\text{pay}_\mathsf{i} = v_\mathsf{i}(\beta, (\text{trade}_j)_{j\in\mathrm{G}})$, for each $\mathsf{i} \in \mathrm{N}$ and $\beta \in \mathcal{B}$
7. $\text{does}(\boldsymbol{\beta}) \rightarrow \bigcirc(\bigwedge_{\mathsf{i}\in\mathrm{N},j\in\mathrm{G}} \text{trade}_{\mathsf{i},j} = \text{WD}^{\varepsilon}_{\lambda_{\mathsf{i},j}}(\boldsymbol{\beta}, \mathsf{X}))$, for each $\boldsymbol{\beta} \in \mathcal{B}^\mathsf{n}$

Figure 3.6: An Iterative Combinatorial Exchange represented by Σ

- $\mathrm{W} = \{\langle(\boldsymbol{\lambda_j})_{j\in\mathrm{G}}, \mathrm{p}, \text{lastbid}\rangle : \text{lastbid}_\mathsf{i} \in \mathcal{B}$ & $\mathrm{p}_\mathsf{i}, \lambda_{\mathsf{i},j} \in \mathrm{I}$ & $\mathsf{i} \in \mathrm{N}$ & $j \in \mathrm{G}\}$;
- $\bar{\mathrm{w}} = \langle 0, \ldots, 0, 0, \ldots, 0, \text{noop}, \ldots, \text{noop}\rangle$;
- $\mathrm{T} = \{\langle(\boldsymbol{\lambda_j})_{j\in\mathrm{G}}, \mathrm{p}, \text{lastbid}\rangle : \text{uncert}(\text{lastbid}_\mathsf{i}) = 0$ & $\text{lastbid}_\mathsf{i} \in \mathcal{B}$ & $\mathrm{p}_\mathsf{i}, \lambda_{\mathsf{i},j} \in \mathrm{I}$ & $\mathsf{i} \in \mathrm{N}$ & $j \in \mathrm{G}\}\}$;
- $\mathrm{L} = \{(\bar{\mathrm{w}}, \mathsf{i}, \beta) : \mathsf{i} \in \mathrm{N}$ & $\beta \in \mathcal{B}\} \cup \{(\langle(\boldsymbol{\lambda_j})_{j\in\mathrm{G}}, \mathrm{p}, \text{lastbid}\rangle, \mathsf{i}, \beta) : \text{eq}(\beta, \text{lastbid}_\mathsf{i})$ & $\text{uncert}(\beta) < \text{uncert}(\text{lastbid}_\mathsf{i})$ & $\beta, \text{lastbid}_\mathsf{r} \in \mathcal{B}$ & $\mathrm{p}_\mathsf{r}, \lambda_{\mathsf{r},j} \in \mathrm{I}$ & $\mathsf{r}, \mathsf{i} \in \mathrm{N}$ & $j \in \mathrm{G}\}$;
- U is defined as follows: for all $\mathrm{w} = \langle(\boldsymbol{\lambda_j})_{j\in\mathrm{G}}, \mathrm{p}, \text{lastbid}\rangle \in \mathrm{W}$ and for all $\boldsymbol{d} \in \mathcal{B}^\mathsf{m}$:
  - If $\mathrm{w} \notin \mathrm{T}$, then $\mathrm{U}(\mathrm{w}, \boldsymbol{d}) = \langle(\boldsymbol{\lambda'_j})_{j\in\mathrm{G}}, \mathrm{p}', \boldsymbol{d}\rangle\rangle$, where each component is updated as follows, for each $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$: $\lambda'_{\mathsf{i},j} = \text{WD}^{\varepsilon}_{\lambda_{\mathsf{i},j}}(\boldsymbol{d}, \mathsf{X})$ and $\mathrm{p}'_\mathsf{i} = v_\mathsf{i}(d_\mathsf{i}, (\boldsymbol{\lambda'_j})_{j\in\mathrm{G}})$.
  - Otherwise, $\mathrm{U}(\mathrm{w}, \boldsymbol{d}) = \mathrm{w}$.
- For each $\mathrm{w} \in \mathrm{W}$, $\mathsf{i} \in \mathrm{N}$ and $j \in \mathrm{G}$, the valuation of numerical variables is as follows: $\pi_\mathrm{Y}(\mathrm{w}, \mathit{trade}_{\mathsf{i},j}) = \lambda_{\mathsf{i},j}$ and $\pi_\mathrm{Y}(\mathrm{w}, \text{pay}_\mathsf{i}) = \mathrm{p}_\mathsf{i}$.

Hereafter, we assume an instance of $\mathrm{M}_{ice} \in \mathcal{M}_{ice}$ and Σ_{ice} for some $\mathcal{B} \subseteq \mathcal{L}_{\mathsf{TBBL}+}$, $\mathrm{I} \subset \mathbb{Z}$, $\mathsf{n}, \mathsf{m} \in \mathrm{I}_{>0}$ and $\mathsf{x}_{\mathsf{i},j} \in \mathrm{I}_{\geq 0}$, where $\mathsf{i} \in \mathrm{N}$, $j \in \mathrm{G}$.

Example 3.3. Let $\mathrm{M}_{ice} \in \mathcal{M}_{ice}$, where (i) there are only two agents, denoted by r and s, (ii) there is only one good type, denoted by a, and (iii) $\mathsf{x}_{\mathsf{r},\mathsf{a}} = 1$ and $\mathsf{x}_{\mathsf{s},\mathsf{a}} = 0$, *i.e*, agent r holds 1 unit of a and agent s has none. Figure 3.5 illustrates a path in M_{ice}. In the initial state w_0, agent r says she wants to sell a for a price between 10€ and 20€ and agent s reports her willingness to buy it for a price between 15€ and

25€. In state w_1, the agents are informed of the provisional trade and payments. Agent s changes her bid to specify she is willing to receive exactly 15€ for selling a. By her turn, agent r specifies a value range 18€ to 20€ for buying a. In w_2, only s can change her bid, since there is no uncertainty in r's bid. Then, s reports her willingness to pay exactly 18€. State w_3 is terminal because there was no uncertainty in the bids reported on w_2. The good is traded and the agents pay (and receive) their asking prices.

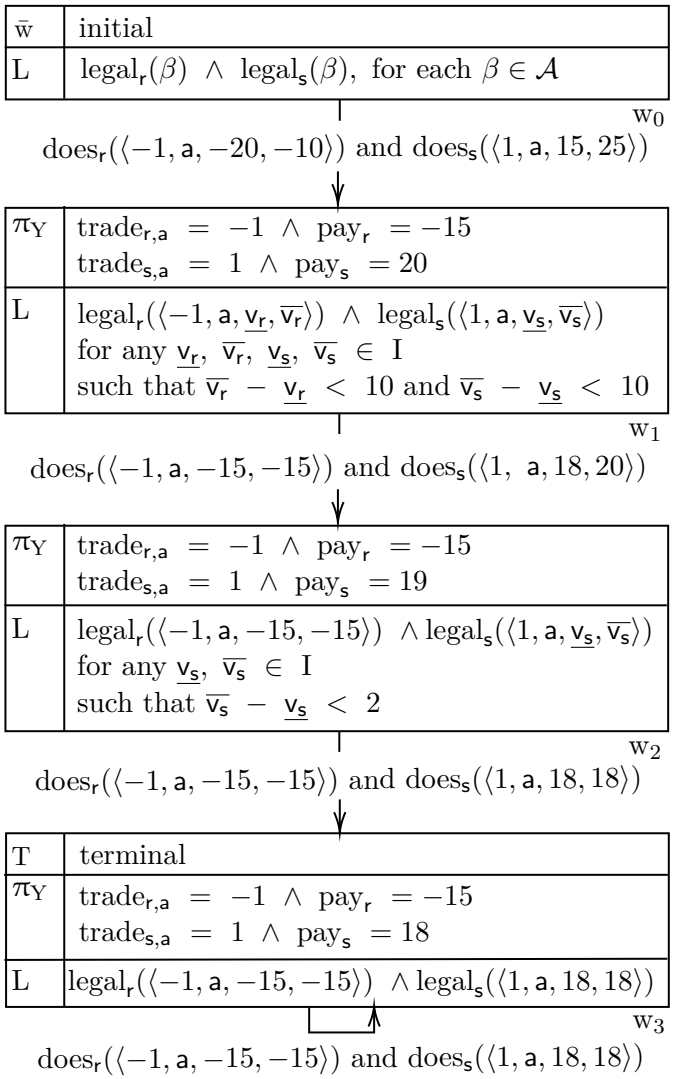

Figure 3.7: A path in M_{ice} where two agents trade a good

3.2.3.2 Evaluating the protocol

Now we focus on the evaluation of Σ_{ice} and M_{ice}. First, Lemma 3.5 shows the soundness of Σ_{ice} over M_{ice}.

Lemma 3.5. M_{ice} *is an ST-model and it is a model of* Σ.

Proof. The proof is similar to those for Lemmas 3.1 and 3.3. □

Since M_{ice} is playable and terminates, Σ_{ice} is well-formed.

Theorem 3.5. Σ_{ice} *is a well-formed protocol over* M_{ice}.

Proof. Since M_{ice} is a model of Σ_{ice}, we show that for each path δ in M_{ice}, each stage $t \geq 0$ in δ and each agent $\mathsf{i} \in \mathrm{N}$,

1. δ is a complete path;
2. $\mathrm{M}_{ice}, \delta, t \models \bigvee_{\beta \in \mathcal{B}} \mathrm{legal}_{\mathsf{i}}(\beta)$.

First, we consider Statement 2. By the definition of $\mathcal{L}_{\mathsf{TBBL}+}$, $\mathrm{uncert}(\beta) \geq 0$ for each bid $\beta \in \mathcal{B}$. According to the legality definition, $\mathrm{uncert}(\theta_{\mathsf{i}}(\delta, t)) < \mathrm{uncert}(\theta_{\mathsf{i}}(\delta, t-1))$ or $\mathrm{uncert}(\theta_{\mathsf{i}}(\delta, t)) = 0$. That is, either the bid reported by i in $\delta[t]$ has less uncertainty than the one she reported in $\delta[t-1]$ or it has no uncertainty. In the case of no uncertainty, $\theta_{\mathsf{i}}(\delta, t) = \theta_{\mathsf{i}}(\delta, e)$ for each $e \geq t$. Since the uncertainty decreases in each turn until being equal to zero, there exists a stage $e \geq 0$ such that $\mathrm{M}_{ice}, \delta, e \models \mathrm{does}_{\mathsf{i}}(\beta) \wedge \mathrm{uncert}(\beta) = 0$ for each agent i. By Rule 5, it follows that the next stage in δ is terminal, that is, $\mathrm{M}_{ice}, \delta, e \models \bigcirc\mathrm{terminal}$.

We now verify Statement 2. If $\delta[t] = \bar{\mathrm{w}}$, then $\mathrm{M}_{ice}, \delta, t \models \mathrm{legal}_{\mathsf{i}}(\beta)$ for each $\beta \in \mathcal{B}$ (Rule 2). Otherwise, $t > 0$ and $\mathrm{M}_{ice}, \delta, t-1 \models \mathrm{does}_{\mathsf{i}}(\beta)$ for some $\beta \in \mathcal{B}$. By the definition of $\mathcal{L}_{\mathsf{TBBL}+}$, $\mathrm{uncert}(\beta) \geq 0$. If $\mathrm{uncert}(\beta) = 0$, then $\mathrm{M}_{ice}, \delta, t \models \mathrm{legal}_{\mathsf{i}}(\beta)$ (Rule 4). Finally, if $\mathrm{uncert}(\beta) < 0$, by the definition of $\mathcal{B}$, there must exist a bid γ such that $\mathrm{eq}(\beta, \gamma)$ and $\mathrm{uncert}(\gamma) < \mathrm{uncert}(\beta)$. According to Rule 3 of Σ_{ice}, $\mathrm{M}_{ice}, \delta, t \models \mathrm{legal}_{\mathsf{i}}(\gamma)$.

□

The cumulative of payments cannot be smaller than zero. However the auction may have positive transfers.

Proposition 3.8. $\mathrm{M}_{ice} \not\models \bigcirc\mathrm{SBB}$ *and* $\mathrm{M}_{ice} \models \bigcirc\mathrm{WBB}$.

Proof. (Sketch) Considering strong budget-balance, notice the path δ illustrated at Figure 3.7 is a counterexample. For instance, in the last stage we have $\mathrm{M}_{ice}, \delta, 3 \models \mathrm{sum}_{\mathsf{i} \in \mathrm{N}}(\mathrm{pay}_{\mathsf{i}}) = 3$ and thus $\mathrm{M}_{ice} \not\models \bigcirc\mathrm{sum}_{\mathsf{i} \in \mathrm{N}}(\mathrm{pay}_{\mathsf{i}}) = 0$ and $\mathrm{M}_{ice} \not\models \bigcirc\mathrm{SBB}$.

Now we consider weak budget-balance. Let δ be a path in M_{ice}, $t \geq 0$ a stage in δ and $\mathsf{i} \in \mathrm{N}$. By the definition of v_{i}, we have that an empty trade is valuated zero, *i.e.*, $v_{\mathsf{i}}(\theta_{\mathsf{i}}(\delta, 0), (0, \ldots, 0)) = 0$. Thus, $\sum_{\mathsf{r} \in \mathrm{N}} v_{\mathsf{r}}(\theta_{\mathsf{r}}(\delta, 0), (0, \ldots, 0)) = 0$. Notice the empty trade $(0, \cdots, 0)$ satisfies Constraints C1-C4 from the winner determination. Among the trades satisfying those constraints, $\mathrm{WD}^{\varepsilon}$ selects the trade $(\boldsymbol{\lambda_j})_{j \in \mathrm{G}}$ that maximizes the cumulative value among all agents, Thus, the cumulative value cannot be smaller than zero. Since the agents' pay the value of their bids, it follows that $\mathrm{M}_{ice}, \delta, t \models \bigcirc\mathrm{sub}_{\mathsf{i} \in \mathrm{N}}(\mathrm{pay}_{\mathsf{i}}) \geq 0$. □

ICE is individually rational, since agents pay their reported preferences.

Proposition 3.9. *Given a joint action* $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$, $\mathrm{M}_{ice} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc \mathrm{IR}(\boldsymbol{\beta})$.

Proof. The proof is straightforward since $\mathrm{M}_{ice} \models \mathrm{does}_{\mathsf{i}}(\beta_{\mathsf{i}}) \rightarrow \bigcirc \mathrm{pay}_{\mathsf{i}} = v_{\mathsf{i}}(\beta_{\mathsf{i}}, (\mathrm{trade}_j)_{j \in \mathrm{G}})$, for each $\mathsf{i} \in \mathrm{N}$ and $\beta_{\mathsf{i}} \in \mathcal{B}$. □

The protocol is efficient since the winner determination maximizes the social welfare given the reported bids.

Proposition 3.10. *Given a joint action* $\boldsymbol{\beta} \in \mathcal{B}^{\mathsf{n}}$, $\mathrm{M}_{ice} \models \mathrm{does}(\boldsymbol{\beta}) \rightarrow \bigcirc \mathrm{EF}(\boldsymbol{\beta})$

Proof. (Sketch) The proof is similar to the proof for Proposition 3.10. □

Since the players' bids influence their payments, they can manipulate the price and the auction is not strategyproof.

Proposition 3.11. M_{ice} *is not strategyproof.*

Proof. (Sketch) We show M_{ice} is not strategyproof with a counterexample. Assume the path δ illustrated in Figure 3.7 and consider stage 2. We have $\theta_{\mathsf{s}}(\delta, 2) = \langle 1, \mathsf{a}, 18, 18 \rangle$. Since the agents pay their reported valuation, in $\delta[3]$, the utility of agent s given her bid is zero, *i.e.*, $\mathrm{M}_{ice}, \delta, 3 \models \mathrm{sub}(v_{\mathsf{s}}(\langle 1, \mathsf{a}, 18, 18 \rangle, (\mathrm{trade}_j)_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{s}}) = 0$. Let δ' be a path in M_{ice} defined exactly like δ, except by the actions performed by s in each stage $t \geq 2$, which is defined as $\theta_{\mathsf{s}}(\delta, t) = \langle 1, \mathsf{a}, 16, 16 \rangle$. Then, $\mathrm{M}, \delta', 3 \models \bigcirc \mathrm{sub}(v_{\mathsf{i}}(\langle 1, \mathsf{a}, 18, 18 \rangle, (\mathrm{trade}_j)_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{s}}) = 2$ and $\mathrm{M}, \delta', 3 \not\models \bigcirc \mathrm{sub}(v_{\mathsf{s}}(\langle 1, \mathsf{a}, 18, 18 \rangle, (\mathrm{trade}_j)_{j \in \mathrm{G}}), \mathrm{pay}_{\mathsf{s}}) \leq 0$. □

3.3 Conclusion

This chapter demonstrated the usefulness of ADL for representing a number of auction-based markets, which include features from single and multi-stage protocols, multiple items, multiple copies of those items and exchange protocols (which generalize double-sided auctions). We also evaluated such auctions in relation to the well-formedness of their ADL-descriptions in addition to economical properties by interpreting them as direct mechanisms.

In the previous chapter, we saw that we can verify most of such properties by model-checking ADL-formulae. Since Algorithm *modelCheck* (see Section 2.4) calls the functions in $\mathcal{F}$ in a polynomial number of times (according to on the formula length), the complexity of computing functions in $\mathcal{F}$ will affect its complexity. For instance, the winner determination problem for combinatorial auctions is NP-hard, which is then the complexity of computing the winner determination functions for TBBL (Lubin et al., 2008). In that case, the model-checking problem for ADL is in Δ_2^{P}, since *modelCheck* consults a NP-oracle a polynomial number of times.

Chapter 4

Actions, Knowledge and Rationality

In the two previous chapters, we focused on the auctioneer perspective by providing tools to specify and verify auction descriptions. In this chapter, we consider the player's perspective and our goal is to show how an agent may reason about their knowledge of other agents' valuations for eliciting her bid in an auction. Specifically, we show that computing a rational bid requires to assume that other agents are also bidding rationally. Following (Aumann, 1995), we understand 'rational' as 'not playing dominated actions'.

The interplay between knowledge, belief and rationality in the context of strategic reasoning is considered in Epistemic Game Theory (Bonanno, 2015; Lorini, 2016). More precisely, this field shows how dominated strategies may be eliminated in an iterative manner (Aumann, 1995). These contributions however require perfect reasoners, who can reason about higher-order knowledge at arbitrary depth, which is unrealistic. Chen et al. (2015) abandon this hypothesis and characterize level-k rationality by means of iterated deletion of strictly dominated strategies. Their approach no longer require perfect reasoners for building strategic bidders considering uncertain information but do not propose a full logic detailing the impact of bounded rationality.

We first extend ADL with knowledge operators from Epistemic GDL (Jiang et al., 2016) and the action modality from the GDL variant proposed in (Zhang and Thielscher, 2015a), which extends GDL with three modalities for representing and reasoning about actions and game strategies. We establish an axiomatic system for the logic and prove its soundness and completeness using variable forgetting technique. We demonstrate how to the logic can be used for creating strategies and verifying properties of strategies

Our extension aims at providing the ground for the design of general auction players. Second, we characterize (bounded) rationality along two dimensions: (i) the impact of the level of higher-order knowledge about other agents and (ii) the impact of looking-ahead beyond the next action to be executed. We also explore the complexity of model-checking for evaluating rationality.

4.1 Epistemic Auction Description Language

In this section, we introduce a logical framework for reasoning about auction protocols while considering imperfect information. The framework is based on a sim-

plified version of ADL (see Section 2.2) and Epistemic GDL (Jiang et al., 2017) and is denoted *Epistemic Auction Description Language* (ADLK).

Definition 4.1. An *auction signature* $\mathcal{S}$ is a tuple $(\mathrm{N}, \mathrm{I}, \mathcal{B}, \Phi, \mathrm{Y})$, where:

- $\mathrm{N} = \{1, 2, \ldots, \mathsf{n}\}$ is a nonempty finite set of *agents*;
- $\mathrm{I} \subset \mathbb{Z}$ is a finite subset of integer numbers representing the range of valuations, bids and payments;
- $\mathcal{B} = \bigcup_{\mathsf{i} \in \mathrm{N}} \mathrm{A}^{\mathsf{i}}$, where each A^{i} consists of a nonempty finite set of *actions* performed by agent $\mathsf{i} \in \mathrm{N}$ and $\mathrm{A}^{\mathsf{i}} \cap \mathrm{A}^{\mathsf{s}} = \emptyset$ if $\mathsf{i} \neq \mathsf{s}$. For convenience, we may write a^{i} for denoting an action in A^{i};
- $\Phi = \{p, q, \ldots\}$ is a finite set of atomic propositions for specifying individual features of a state;
- $\mathrm{Y} = \{y_1, y_2, \ldots\}$ is a finite set of numerical variables for specifying numerical features of a state.

We assume a total order among the agents in N, denoted by $\prec$, where $r \prec i$ means that agent i precedes agent i in $\prec$; it will be used to break ties in winner determination. Throughout the rest of this section, we fix an auction signature $\mathcal{S}$ and all concepts will be based on this signature, except if stated otherwise. We still consider a semantics based on state-transition models. Differently from the ST-models presented on Chapter 2, these models consider a set of initial states and contain equivalence relations for indicating states that are indistinguishable for each agent.

Definition 4.2. An epistemic state transition ET-*model* M is a tuple $(\mathrm{W}, \mathrm{W}_\iota, \mathrm{T}, \{\mathrm{R}_{\mathsf{i}}\}_{\mathsf{i} \in \mathrm{N}}, \mathrm{U}, \pi_\Phi, \pi_\mathrm{Y})$, where:

- W is a finite nonempty set of *states*;
- $\mathrm{W}_\iota \subseteq \mathrm{W}$ is a set of *initial* states;
- $\mathrm{T} \subseteq \mathrm{W} \setminus \mathrm{W}_\iota$ is a set of *terminal* states;
- $\mathrm{R}_{\mathsf{i}} \subseteq \mathrm{W} \times \mathrm{W}$ is an equivalence relation for agent i, indicating the states that are indistinguishable for i;
- $\mathrm{U} : \mathrm{W} \times (\prod_{\mathsf{i} \in \mathrm{N}} \mathrm{A}^{\mathsf{i}}) \to \mathrm{W}$ is an *update* function, specifying the transitions for each combination of joint actions;
- $\pi_\Phi : \mathrm{W} \to 2^\Phi$ is the valuation function for the state propositions;
- $\pi_\mathrm{Y} : \mathrm{W} \times \mathrm{Y} \to \mathrm{I}$ is the valuation function for the numerical variables.

For a group of agents $\mathrm{C} \in 2^{\mathrm{N}} \setminus \{\emptyset\}$, we write $d^{\mathrm{C}} \in \prod_{\mathsf{i}\in\mathrm{C}} \mathrm{A}^{\mathsf{i}}$ to denote a joint action of the agents in C. We denote by d^{i} the individual action for agent $\mathsf{i} \in \mathrm{C}$ in the joint action d^{C}. When $\mathrm{C} = \mathrm{N}$ then we omit N and simply write d instead of d^{N}. Let $\mathrm{R}_{\mathsf{i}}(\mathrm{w})$ denote the set of all states that agent i cannot distinguish from w, *i.e.*, $\mathrm{R}_{\mathsf{i}}(\mathrm{w}) = \{\mathrm{v} \in \mathrm{W} : \mathrm{w}\mathrm{R}_{\mathsf{i}}\mathrm{v}\}$.

Unlike the previous chapters, we adopt a notion of *moves* instead of paths. Our notion of move resembles the turn-based definition proposed by Zhang and Thielscher (2015a,b). For every $\mathrm{w} \in \mathrm{W}$ and $d \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}$, we call (w, d) a *move*. Given a group of agents $\mathrm{C} \in 2^{\mathrm{N}} \setminus \{\emptyset\}$, we write $(\mathrm{w}, \langle d^{\mathrm{C}}, d^{\text{-}\mathrm{C}}\rangle)$ instead of (w, d) when we want to talk about C's part in (w, d), where $d^{\text{-}\mathrm{C}} \in \prod_{\mathsf{s}\in\mathrm{N}\setminus\mathrm{C}} \mathrm{A}^{\mathsf{s}}$ denotes the actions of all the agents except those in C in the joint action d.

Definition 4.3. Two moves (w, d) and (v, e) are equivalent for agent i, written $(\mathrm{w}, d) \approx_{\mathsf{i}} (\mathrm{v}, e)$, iff $\mathrm{w}\mathrm{R}_{\mathsf{i}}\mathrm{v}$ and $d^{\mathsf{i}} = e^{\mathsf{i}}$.

Clearly relation $\approx_{\mathsf{i}}$ is reflexive, transitive and symmetric. Differently from standard GDL, our semantics is based on moves instead of paths. This allows the agent to reason about the effects of actions without exploring all ways the game could proceed (*i.e.*, all the reachable states in each complete path where she takes this action). In ADLK, we define the action execution modality in games with synchronous moves. The idea of move-based semantics and action modalities stems from the proposal of Zhang and Thielscher (2015a). Their approach is restricted to turn-based games, where only one action can be performed at a given state.

4.1.1 Syntax

First, we introduce ADLK syntax, a simplified version of ADL with epistemic operators and action modality. Let $z \in \mathcal{L}_Z$ be a numerical term defined as follows:

$$z ::= t \mid \mathrm{sum}(z, z) \mid \mathrm{sub}(z, z) \mid \mathrm{min}(z, z) \mid \mathrm{max}(z, z) \mid \mathrm{times}(z, z) \mid y$$

where $t \in \mathrm{I}$, $y \in Y$. The meaning of numerical terms is the natural one; for instance, the term $\mathrm{min}(z_1, z_2)$ specifies the minimum value between z_1 and z_2. Finally, y denotes the value of the variable $y \in \mathrm{Y}$ in the current state.

A formula in ADLK, denoted $\varphi \in \mathcal{L}_{\mathsf{ADLK}}$, is defined by the following BNF:

$$\varphi ::= p \mid z \otimes z \mid r \prec r \mid \mathrm{initial} \mid \mathrm{terminal} \mid \mathrm{does}(a^{\mathsf{i}}) \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{K}_{\mathsf{i}}\varphi \mid [\,d^{\mathrm{C}}\,]\varphi$$

where $p \in \Phi$, $\mathsf{i} \in \mathrm{N}$, $\otimes \in \{>, <, =\}$, $a^{\mathsf{i}} \in \mathcal{B}$, $\mathrm{C} \in 2^{\mathrm{N}} \setminus \{\emptyset\}$, $d^{\mathrm{C}} \in \prod_{\mathsf{i}\in\mathrm{C}} \mathrm{A}^{\mathsf{i}}$ and $z \in \mathcal{L}_Z$. Other connectives $\vee, \rightarrow, \leftrightarrow, \top$ and $\bot$ are defined by $\neg$ and $\wedge$ in the standard way. The comparison operators $\leq$, $\geq$ and $\neq$ are defined by $\vee, >, <$ and $=$. The extension of the operators $>, <$ and $=$ and numerical terms $\mathrm{max}(z_1, z_2), \mathrm{min}(z_1, z_2)$, $\mathrm{sum}(z_1, z_2)$ to multiple arguments is straightforward. The formula $r_1 \prec r_2$ denotes the tie-breaking priority of r_1 over r_2.

Intuitively, initial and terminal specify the initial and the terminal states, respectively; $\mathrm{does}(a^{\mathsf{i}})$ asserts that agent i takes action a^{i} at the current move. The

epistemic operator K_i is taken from the Epistemic Logic (Fagin et al., 2003). The formula $\mathsf{K}_\mathsf{i}\varphi$ is read as "agent i knows that φ". The action execution operator comes from the GDL variant with action modalities (Zhang and Thielscher, 2015a) and the formula $[\,d^{\mathrm{C}}\,]\varphi$ means that if joint action d^{C} is executed, φ will be true next. The abbreviation does(d^{C}) specifies that each agent in C performs her respective action in d^{C}, that is,

$$\mathrm{does}(d^{\mathrm{C}}) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{C}} \mathrm{does}(d^{\mathsf{i}})$$

As Zhang and Thielscher (2015a), we use the action modality to define the temporal operator $\bigcirc$:

$$\bigcirc\varphi =_{def} \bigvee_{d\in\prod_{\mathsf{i}\in\mathrm{N}}\mathrm{A}^{\mathsf{i}}} (\mathrm{does}(d) \wedge [\,d\,]\varphi)$$

The formula $\bigcirc\varphi$ reads "φ will be true next". We also use the following abbreviation from Epistemic Logic: $\widehat{\mathsf{K}}_\mathsf{i}\varphi =_{def} \neg\mathsf{K}_\mathsf{i}\neg\varphi$ where $\widehat{\mathsf{K}}_\mathsf{i}\varphi$ represents that "φ is compatible with agent i's knowledge". Given $j > 0$ and $\mathrm{C} \in 2^{\mathrm{N}} \setminus \{\emptyset\}$, we write $\sigma^{\mathrm{C}} = (\prod_{\mathsf{i}\in\mathrm{C}} \mathrm{A}^{\mathsf{i}})^j$ for a sequence of joint actions for C. The i-th joint action in σ^{C} is noted σ^{C}_i. Finally, define $[\,\sigma^{\mathrm{C}}\,]^j\,\varphi$, for $|\sigma^{\mathrm{C}}| = j$ by induction of j:

$$[\,\sigma^{\mathrm{C}}\,]^1\,\varphi =_{\mathrm{def}} [\,\sigma^{\mathrm{C}}\,]\varphi$$
$$[\,\sigma^{\mathrm{C}}\,]^{j+1}\,\varphi =_{\mathrm{def}} [\,\sigma^{\mathrm{C}}\,][\,\sigma^{\mathrm{C}}_j\,]\varphi$$

The formula $[\,\sigma^{\mathrm{C}}\,]^j\,\varphi$ means that if the group C followed the sequence of joint actions described by σ^{C} for the next j stages, then φ would hold.

4.1.2 Semantics

The semantics for ADLK is given in two steps. First, function f interprets the meaning of numerical terms $z \in \mathcal{L}_Z$. Next, a formula $\varphi \in \mathcal{L}_{\mathsf{ADLK}}$ is interpreted with respect to a move. In Definition 4.4, we specify function f to evaluate the meaning of any $z \in \mathcal{L}_Z$ in a move.

Definition 4.4. Let M be an ET-Model. Define Function $f : W \times (\prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}) \times \mathcal{L}_Z \to \mathbb{Z}$, assigning any $\mathrm{w} \in \mathrm{W}$, $d \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}$, and $z \in \mathcal{L}_Z$ to a number in $\mathbb{Z}$:

If z is on the form $\mathrm{sum}(z', z'')$, $\mathrm{sub}(z', z'')$, $\mathrm{min}(z', z'')$, $\mathrm{max}(z', z'')$ or $\mathrm{times}(z', z'')$, then $f(\mathrm{w}, d, z)$ is defined through the application of the corresponding mathematical operators and functions over $f(\mathrm{w}, d, z')$ and $f(\mathrm{w}, d, z'')$. Otherwise, $f(\mathrm{w}, d, z) = z$ if $z \in \mathrm{I}$ and $f(\mathrm{w}, d, z) = \pi_Y(\mathrm{w}, z)$ if $z \in Y$.

Definition 4.5. Let M be an ET-Model. Given a move (w, d), where $\mathrm{w} \in \mathrm{W}$ and $d \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}$, and a formula $\varphi \in \mathcal{L}_{ADL}$, we say that φ is true in the move (w, d)

under M, denoted by $M \models_{(\mathrm{w},d)} \varphi$, according to the following rules:

$$
\begin{array}{ll}
\mathrm{M} \models_{(\mathrm{w},d)} p & \text{iff } p \in \pi_\Phi(\mathrm{w}) \\
\mathrm{M} \models_{(\mathrm{w},d)} \neg\varphi & \text{iff } \mathrm{M} \not\models_{(\mathrm{w},d)} \varphi \\
\mathrm{M} \models_{(\mathrm{w},d)} \varphi_1 \wedge \varphi_2 & \text{iff } \mathrm{M} \models_{(\mathrm{w},d)} \varphi_1 \text{ and } \mathrm{M} \models_{(\mathrm{w},d)} \varphi_2 \\
\mathrm{M} \models_{(\mathrm{w},d)} \text{initial} & \text{iff } \mathrm{w} \in \mathrm{W}_\iota \\
\mathrm{M} \models_{(\mathrm{w},d)} \text{terminal} & \text{iff } \mathrm{w} \in \mathrm{T} \\
\mathrm{M} \models_{(\mathrm{w},d)} r_1 \prec r_2 & \text{iff } r_1 \prec r_2 \\
\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{does}(a^{\mathsf{i}}) & \text{iff } d^{\mathsf{i}} = a^{\mathsf{i}} \\
\mathrm{M} \models_{(\mathrm{w},d)} z_1 \otimes z_2 & \text{iff } f(w,d,z_1) \otimes f(w,d,z_2), \text{where } \otimes \in \{>,<,=\} \\
\mathrm{M} \models_{(\mathrm{w},d)} \mathsf{K}_{\mathsf{i}}\varphi & \text{iff for every } \mathrm{v} \in \mathrm{W} \text{ and } e \in \prod_{\mathsf{s}\in\mathrm{N}} \mathrm{A}^{\mathsf{s}}, \text{ if } (\mathrm{w},d) \approx_{\mathsf{i}} (\mathrm{v},e), \\
 & \quad \text{then } \mathrm{M} \models_{(\mathrm{v},e)} \varphi \\
\mathrm{M} \models_{(\mathrm{w},d)} [\, b^{\mathrm{C}} \,]\varphi & \text{iff } \mathrm{M} \models_{(U(\mathrm{w},e),c)} \varphi, \text{ where } e = \langle b^{\mathrm{C}}, d^{-\mathrm{C}} \rangle, \text{ for every} \\
 & \quad c \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}
\end{array}
$$

A formula φ is *globally true* in an ET-Model M, written $\mathrm{M} \models \varphi$, if $\mathrm{M} \models_{(\mathrm{w},d)} \varphi$ for all $\mathrm{w} \in \mathrm{W}$ and $d \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}$. Finally, let Σ be a set of formulas in $\mathcal{L}_{\mathsf{ADLK}}$, then M is a *model* of Σ if $M \models \varphi$ for all $\varphi \in \Sigma$.

Each K_{i} is a normal modal operator. It satisfies that if all i-accessible worlds agree on φ then i knows either φ or $\neg\varphi$. If φ is true then i knows that φ.

Proposition 4.1. *Let* M *be an ET-Model,* $\mathsf{i} \in \mathrm{N}$ *be an agent and* $\varphi \in \mathcal{L}_{\mathsf{ADLK}}$ *be a formula, then* $\mathrm{M} \models \varphi \rightarrow \mathsf{K}_{\mathsf{i}}\varphi$ *if and only if for all* $\mathrm{w}, \mathrm{v} \in \mathrm{W}$ *and all* $d, e \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}$ *such that* $(\mathrm{w},d) \approx_{\mathsf{i}} (\mathrm{v},e)$, $\mathrm{M} \models_{(\mathrm{w},d)} \varphi$ *iff* $\mathrm{M} \models_{(\mathrm{v},e)} \varphi$.

It follows from the equivalence relation $\approx_{\mathsf{i}}$ that agent i knows the actions she performs. This is similar to the uniform strategies in Alternating-time Temporal Epistemic Logic (Jamroga and van der Hoek, 2004) and Dynamic Epistemic Logic (Van Benthem, 2001).

Proposition 4.2. *For any agent* $\mathsf{i} \in \mathrm{N}$, *action* $a^{\mathsf{i}} \in \mathrm{A}^{\mathsf{i}}$, *formula* $\varphi \in \mathcal{L}_{\mathsf{ADLK}}$, *number of steps* $j > 0$, *group of agents* $\mathrm{C} \in 2^{\mathrm{N}} \setminus \{\emptyset\}$ *and* $\sigma^{\mathsf{i}} \in (\prod_{\mathsf{i}\in\mathrm{C}} \mathrm{A}^{\mathsf{i}})^j$*:*

1. $\mathrm{M} \models \mathrm{does}(a^{\mathsf{i}}) \rightarrow \mathsf{K}_{\mathsf{i}}\mathrm{does}(a^{\mathsf{i}})$
2. *If* $\mathrm{M} \models [\, \sigma^{\mathrm{C}} \,]^j \varphi$ *then* $\mathrm{M} \models \mathsf{K}_{\mathsf{i}}[\, \sigma^{\mathrm{C}} \,]^j \varphi$
3. *If* $\mathrm{M} \models [\, \sigma^{\mathrm{C}} \,]^j \mathsf{K}_{\mathsf{i}}\varphi$ *then* $\mathrm{M} \models \mathsf{K}_{\mathsf{i}}[\, \sigma^{\mathrm{C}} \,]^j \varphi$

Let us now illustrate how to represent an auction-based protocol in ADLK, namely, a Dutch auction. First, we show the syntactical representation through ADLK-formulas. Later, we address the semantical representation.

4.1.3 Dutch Auction with Private Valuations

In a Dutch auction, the auctioneer starts by proposing a high asking price. The price is decreased until it reaches a predefined reserve price or some bidder shows interest at purchasing the good. The auction then ends and the object is sold at the given price to the bidder who signaled her interest (Krishna, 2009).

Let $\mathcal{S}_{\text{dut}}$ be an auction signature and $\mathsf{starting}, \mathsf{reserve} \in \mathbb{N}$, $\mathsf{dec}, \mathsf{n} \in \mathbb{N} \setminus \{0\}$ be constant values. The constants $\mathsf{starting}, \mathsf{reserve}$, dec, n represent the starting and reserve prices, the decrement in each round and the number of agents, respectively. The auction signature is defined as follows: $\mathcal{S}_{\text{dut}} = (\mathrm{N}, \mathrm{I}, \mathcal{B}, \Phi, \mathrm{Y})$, where $\mathrm{N} = \{1, \ldots, \mathsf{n}\}$, $\mathrm{I} = \{0, \ldots, \mathsf{starting}\}$, $\mathcal{B} = \{\text{bid}_\mathsf{i}, \text{wait}_\mathsf{i} : \mathsf{i} \in \mathrm{N}\}$, $\Phi = \{\text{winner}_\mathsf{i} : \mathsf{i} \in \mathrm{N}\}$ and $\mathrm{Y} = \{\text{pay}_\mathsf{i}, \vartheta_\mathsf{i} : \mathsf{i} \in \mathrm{N}\}$. The numerical variables pay_i and ϑ_i specify the payment and the private valuation for an agent i.

4.1.3.1 Syntactical Representation

The rules of the Dutch auction are formulated by ADLK-formulas as shown in Figure 4.1.

1. $\text{initial} \leftrightarrow \text{price} = \mathsf{starting} \wedge \bigwedge_{\mathsf{i} \in \mathrm{N}} \neg\text{winner}_\mathsf{i}$
2. $\text{winner}_\mathsf{i} \rightarrow \text{pay}_\mathsf{i} = \text{price}$, for each $\mathsf{i} \in \mathrm{N}$
3. $\neg\text{winner}_\mathsf{i} \rightarrow \text{pay}_\mathsf{i} = 0$, for each $\mathsf{i} \in \mathrm{N}$
4. $\text{terminal} \leftrightarrow \text{sub}(\text{price}, \mathsf{dec}) < \mathsf{reserve} \vee \bigvee_{\mathsf{i} \in \mathrm{N}} \text{winner}_\mathsf{i}$
5. $\neg\text{terminal} \wedge \text{price} = x \wedge \bigwedge_{\mathsf{i} \in \mathrm{N}} \text{does}(\text{wait}_\mathsf{i}) \rightarrow \bigcirc(\text{price} = \text{sub}(\text{price}, \mathsf{dec}) \wedge \bigwedge_{\mathsf{i} \in \mathrm{N}} \neg\text{winner}_\mathsf{i})$, for each $x \in \mathrm{I}$
6. $\neg\text{terminal} \wedge \text{price} = x \wedge \text{does}(\text{bid}_\mathsf{i}) \wedge \bigwedge_{\mathsf{s} \neq \mathsf{i}, \mathsf{s} \in \mathrm{N}} (\neg\text{does}(\text{bid}_\mathsf{s}) \vee \mathsf{i} \prec \mathsf{s}) \rightarrow \bigcirc(\text{winner}_\mathsf{i} \wedge \bigwedge_{\mathsf{s} \neq \mathsf{i}, \mathsf{s} \in \mathrm{N}} \neg\text{winner}_\mathsf{s})$, for each $x \in \mathrm{I}$ and each $\mathsf{i} \in \mathrm{N}$
7. $\text{terminal} \wedge y = x \rightarrow \bigcirc y = x$, for each $y \in \mathrm{Y}$ and each $x \in \mathrm{I}$
8. $\text{terminal} \wedge \mathit{win} \rightarrow \bigcirc \mathit{win}$, for each $\mathit{win} \in \{\text{winner}_\mathsf{i}, \neg\text{winner}_\mathsf{i} : \mathsf{i} \in \mathrm{N}\}$
9. $\mathsf{K}_\mathsf{i}(\vartheta_\mathsf{i} = x) \vee \mathsf{K}_\mathsf{i}\neg(\vartheta_\mathsf{i} = x)$, for each $x \in \mathrm{I}$ and $\mathsf{i} \in \mathrm{N}$

Figure 4.1: Dutch auction represented by Σ_{dut}

In an initial state, the price starts at $\mathsf{starting}$ and there is no winner (Rule 1). If an agent is a winner, she pays the current price. Otherwise, she does not pay anything (Rules 2 and 3). The terminal state is reached when it is not possible to decrease the price anymore or there is a winner (Rule 4). While not in the terminal state, the price either decreases if no agent bids or the price is settled if some agent accepted to purchase the good (Rules 5 and 6). If only one agent accepts, she is marked as the winner. In case two or more agents bid, the winner is assigned

according to the tie-breaking rule. Rules 7 and 8 ensure no proposition or numerical variable change its value after a terminal state. Finally, Rule 9 specifies that each agent is aware of how much she valuates the good. Let Σ_{dut} be the set of Rules 1-9.

4.1.3.2 Model Representation

Let us address the model representation of the Dutch auction. Let us define $\mathcal{M}_{\text{dut}}$ as the class of models M_{dut} defined for a signature $\mathcal{S}_{\text{dut}}$ and the constants $\mathsf{starting}, \mathsf{reserve}, \mathsf{dec}$ and n. Each $\text{M}_{\text{dut}} = (\text{W}, \text{W}_\iota, \text{T}, \{\text{R}_\mathsf{i}\}_{\mathsf{i}\in\text{N}}, \text{U}, \pi_\Phi, \pi_\text{Y})$ is defined as follows:

- $\text{W} = \{\langle \text{pr}, \text{buyer}, \theta_1, \ldots, \theta_\mathsf{n}\rangle : 0 \leq \text{pr} \leq \mathsf{starting}$ & $\text{buyer} \in \text{N} \cup \{none\}$ & $0 \leq \text{val}_\mathsf{i} \leq \mathsf{starting}$ for each $\mathsf{i} \in \text{N}\}$;
- $\text{W}_\iota = \{\langle \mathsf{starting}, none, \text{val}_1, \ldots, \text{val}_\mathsf{n}\rangle : 0 \leq \text{val}_\mathsf{i} \leq \mathsf{starting}$ for each $\mathsf{i} \in \text{N}\}$;
- $\text{T} = \{\langle \text{pr}, \text{buyer}, \theta_1, \ldots, \theta_\mathsf{n}\rangle : 0 \leq \text{pr} \leq \mathsf{starting}$ & $\text{buyer} \in \text{N}$ & $0 \leq \text{val}_\mathsf{i} \leq \mathsf{starting}$ for each $\mathsf{i} \in \text{N}\} \cup \{\langle \text{pr}, \text{buyer}, \theta_1, \ldots, \theta_\mathsf{n}\rangle : \text{pr} - \mathsf{dec} < \mathsf{reserve}$ & $\text{buyer} \in \text{N} \cup \{none\}$ & $0 \leq \text{val}_\mathsf{i} \leq \mathsf{starting}$ for each $\mathsf{i} \in \text{N}\}$;
- For each agent $\mathsf{i} \in \text{N}$ and for any two states $\text{w} = \langle \text{pr}, \text{buyer}, \theta_1, \ldots, \theta_\mathsf{n}\rangle$ and $\text{v} = \langle \text{pr}', \text{buyer}', \text{val}'_1, \ldots, \text{val}'_\mathsf{n}\rangle$ in W, the relation R_i is defined as follows: $\text{w}\text{R}_\mathsf{i}\text{v}$ iff (i) $\text{pr} = \text{pr}'$; (ii) $\text{buyer} = \text{buyer}'$; and (iii) $\text{val}_\mathsf{i} = \text{val}'_\mathsf{i}$.
- For all states $\text{w} = \langle \text{pr}, \text{buyer}, \theta_1, \ldots, \theta_\mathsf{n}\rangle$ and all joint actions $d = (a^\mathsf{i})_{\mathsf{i}\in\text{N}}$, such that $\text{w} \in \text{W}$ and $a^\mathsf{i} \in \{\text{bid}_\mathsf{i}, \text{wait}_\mathsf{i}\}$, we define U as follows:
  - If $\text{w} \notin \text{T}$, then $\text{U}(\text{w}, d) = \langle \text{pr}', \text{buyer}', \theta_1, \ldots, \theta_\mathsf{n}\rangle$, such that the components pr' and buyer' are defined as follows: (i) $\text{pr}' = \text{pr} - \mathsf{dec}$ if $a^\mathsf{i} = \text{wait}_\mathsf{i}$, for all $\mathsf{i} \in \text{N}$; otherwise $\text{pr}' = \text{pr}$; (ii) $\text{buyer}' = \mathsf{i}$ if $a^\mathsf{i} = \text{bid}_\mathsf{i}$ for some $\mathsf{i} \in \text{N}$ and for all $\mathsf{s} \in \text{N}$ such that $\mathsf{s} \neq \mathsf{i}$, either $a^\mathsf{s} = \text{wait}_\mathsf{s}$ or $\mathsf{i} \prec \mathsf{s}$; otherwise, $\text{buyer}' = none$;
  - Otherwise, $\text{U}(\text{w}, d) = \text{w}$.
- Finally, for each state $\text{w} = \langle \text{pr}, \text{buyer}, \theta_1, \ldots, \theta_\mathsf{n}\rangle$, such that $\text{w} \in \text{W}$, let $\pi_\Phi(\text{w}) = \{\text{winner}_\mathsf{i} : \text{buyer} = \mathsf{i}$ & $\mathsf{i} \in \text{N}\}$; $\pi_\text{Y}(\text{w}, \text{price}) = \text{pr}$. For each agent $\mathsf{i} \in \text{N}$, let $\pi_\text{Y}(\text{w}, \vartheta_\mathsf{i}) = \text{val}_\mathsf{i}$ and $\pi_\text{Y}(\text{w}, \text{pay}_\mathsf{i}) = \text{pr}$ if $\text{buyer} = \mathsf{i}$. Otherwise, $\pi_\text{Y}(\text{w}, \text{pay}_\mathsf{i}) = 0$.

Let us assume a model $\text{M}_{\text{dut}} \in \mathcal{M}_{\text{dut}}$ and Σ_{dut} for some $\mathcal{S}_{\text{dut}}$ and the constants $\mathsf{starting}, \mathsf{reserve} \in \mathbb{N}$, $\mathsf{dec}, \mathsf{n} \in \mathbb{N} \setminus \{0\}$.

Proposition 4.3. M_{dut} *is an ET-Model and* $\text{M}_{\text{dut}} \models \Sigma_{\text{dut}}$, i.e., M_{dut} *is a model of* Σ_{dut}.

That is, M_{dut} is a sound representation of Σ_{dut}. Notice that as M_{dut} is not the unique model for Σ_{dut}, thereby, the completeness does not hold. It follows from Prop. 4.1 and 4.3 that each agent knows the auction rules denoted by Σ_{dut}, that is, $\text{M}_{\text{dut}} \models \bigwedge_{\mathsf{i}\in\text{N}}(\mathsf{K}_\mathsf{i}\Sigma_{\text{dut}})$. In the next section, we define rationality in ADLK.

4.2 Rationality in Auctions

To characterize rationality of auction players, we assume $\{\vartheta_{\mathsf{i}}, \text{pay}_{\mathsf{i}} : \mathsf{i} \in \text{N}\} \subseteq Y$ and $\{\text{winner}_{\mathsf{i}} : \mathsf{i} \in \text{N}\} \subseteq \Phi$, where $\vartheta_{\mathsf{i}}, \text{pay}_{\mathsf{i}}$ and $\text{winner}_{\mathsf{i}}$ specify the agents valuation, payment and whether she won the auction, resp. Let $ut \in \text{I}$, we denote whether the utility of agent $\mathsf{i} \in \text{N}$ is equal to ut in a single good and unit auction according to the truth value of the following formula:

$$\text{utility}_{\mathsf{i}} = ut =_{\text{def}} (ut = \text{sub}(\vartheta_{\mathsf{i}}, \text{pay}_{\mathsf{i}}) \wedge \text{winner}_{\mathsf{i}}) \vee (ut = -\text{pay}_{\mathsf{i}} \wedge \neg\text{winner}_{\mathsf{i}})$$

Note that we can extend the notion of utility to multiple units and goods by including numerical variables representing the agents' allocations and their valuations for such allocations. In this work, we focus on epistemic reasoning about action choice and rationality of auction players.

Similar to the strong strategy dominance (Lorini, 2016), we say an action a^{i} of an agent i is a *strongly dominated action* if and only if, there exists another action b^{i} of i such that, for all actions $a^{-\mathsf{i}}$ of the other agents, playing b^{i} while others play $a^{-\mathsf{i}}$ leads to a better utility than playing a^{i} while others play $a^{-\mathsf{i}}$. In ADLK, the agents' utility is captured in a move of a model and the action choice operator allows us to compare what would have happened if a group of agents took a given joint action.

4.2.1 Bounded Rationality

We adapt the weak rationality formalization by Lorini (2016) to ADLK formulas. Different from his approach, we consider levels of rationality instead of common knowledge. Our notion of k-order rationality is based on (Chen et al., 2015): an agent is k-order rational if she is weakly rational and knows all agents are $(k-1)$-order rational.

GDL-based languages explicit the stages of a game execution through paths (or runs). The game starts from an initial state and the succeeding states are defined according to the agents' joint actions. Since GDL agents choose "on-the-fly strategies" during the game, the players should be able to evaluate the current state of the game and to decide which action they will execute.

Adopting these features from GDL in ADLK allows us to explicitly model information feedback, which is a key feature in the design of iterative auctions (Parkes, 2006). For instance, in ADLK, we can describe auctions where the agents are assigned to allocations and payments at any stage, which may be different from their final assignments in the terminal state. For this reason, instead of defining utilities as a function to strategy profiles as in ATL (Alur et al., 2002), we model the agents' utility as being dependent on the current state of the auction.

We rephrase the rationality notions proposed by Chen et al. (2015) and Lorini (2016). Our definitions generalize such notions by, at first, considering k-order of knowledge and, second, by taking into account state-based utilities and exploring bounded sequences of actions. A rational agent plays according to her utility after

performing an action. When reasoning about iterative auctions, the agent considers her utility after playing according to a sequence of j actions. Since most auction-based markets are finite (in the sense that the auction finishes eventually), it is reasonable to assume the agents only need to include in their reasoning which actions may occur in the next j steps. Given a fixed number of steps $j > 0$, we inductively define that an agent is k-order rational, for $k \leq j$. The base case is that any agent is 0-order rational, that is, $\mathrm{Rat}(\mathsf{i}, 0, j) =_{\mathrm{def}} \top$. For all $k > 0$, we define:

$$\mathrm{Rat}(\mathsf{i}, k+1, j) =_{\mathrm{def}} \mathrm{WR}(\mathsf{i}, j) \wedge \mathsf{K}_{\mathsf{i}}(\bigwedge_{\mathsf{s} \in \mathrm{N}} \mathrm{Rat}(\mathsf{s}, k, j))$$

That is, an agent is $(k+1)$-order rational if she is weakly rational when looking j stages ahead and knows every other agent is k rational. Weak rationality is defined by:

$$\mathrm{WR}(\mathsf{i}, j) =_{\mathrm{def}} \bigwedge_{a^{\mathsf{i}} \in \mathrm{A}^{\mathsf{i}}} (\mathrm{does}(a^{\mathsf{i}}) \rightarrow \bigvee_{\rho^{\mathsf{i}} \in (\mathrm{A}^{\mathsf{i}})^{j-1}} \mathrm{WRAction}(\mathsf{i}, (a^{\mathsf{i}}, \rho^{\mathsf{i}}), j))$$

where

$$\mathrm{WRAction}(\mathsf{i}, \sigma^{\mathsf{i}}, j) =_{def} \bigwedge_{\chi^{\mathsf{i}} \in (\mathrm{A}^{\mathsf{i}})^j} \Big(\bigvee_{\sigma^{\text{-}\mathsf{i}} \in (\prod_{\mathsf{s} \neq \mathsf{i}} \mathrm{A}^{\mathsf{s}})^j} (\widehat{\mathsf{K}}_{\mathsf{i}} \mathrm{does}(\sigma_1^{\text{-}\mathsf{i}}) \wedge \bigvee_{ut, ut' \in \mathrm{I}} ([\, \chi^{\mathsf{i}}, \sigma^{\text{-}\mathsf{i}} \,]^j \, \mathrm{utility}_{\mathsf{i}} = ut' \wedge [\, \sigma^{\mathsf{i}}, \sigma^{\text{-}\mathsf{i}} \,]^j \, \mathrm{utility}_{\mathsf{i}} = ut \wedge ut' \leq ut)) \Big)$$

An agent a^{i} is weakly rational when reasoning j stages ahead if when she performs an action a^{i}, there exists a sequence of j actions starting by a^{i} that is weakly rational for her to follow over j stages. Finally, it is weakly rational for agent i to follow a sequence of actions σ^{i} for j steps, noted $\mathrm{WRAction}(\mathsf{i}, \sigma^{\mathsf{i}}, j)$, if for every other sequence of actions χ^{i} there exists a sequence of joint actions $\sigma^{\text{-}\mathsf{i}}$ that i considers possible to be executed such that her utility after following σ^{i} for j steps is at least as good as her utility after following χ^{i}.

Notice that if j is large enough to reach terminal states, the state-based utilities represent strategy-based utility functions. Our definition of rationality requires to assume that all agents are rational: as soon as one is known to be non-rational, it is no longer possible to be k-order rational, for $k > 1$. This requirement entails that looking ahead without considering knowledge leads to consider all actions as rational:

Proposition 4.4. *For every ET-Model* M*, state* $\mathrm{w} \in \mathrm{W}$*, joint action* $d \in \prod_{\mathsf{i} \in \mathrm{N}} \mathrm{A}^{\mathsf{i}}$*, agent* $\mathsf{i} \in \mathrm{N}$ *and* $j > 0$*, it holds that* $\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{does}(d^{\mathsf{i}}) \wedge \mathrm{Rat}(\mathsf{i}, 0, j)$.

Next, considering higher-order knowledge enables us to eliminate strongly dominated actions.

Theorem 4.1. *For any ET-Model* M*, state* $\mathrm{w} \in \mathrm{W}$*, joint action* $d \in \prod_{\mathsf{i} \in \mathrm{N}} \mathrm{A}^{\mathsf{i}}$*,* $k > 0$*,* $j > 0$*, agent* $\mathsf{i} \in \mathrm{N}$ *and action* $a^{\mathsf{i}} \in \mathrm{A}^{\mathsf{i}}$*, if* $\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{does}(a^{\mathsf{i}}) \wedge \mathrm{Rat}(\mathsf{i}, k, j)$ *then* $\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{does}(a^{\mathsf{i}}) \wedge \mathrm{Rat}(\mathsf{i}, k-1, j)$.

Proof. Assume $\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{does}(a^{\mathsf{i}}) \wedge \mathrm{Rat}(\mathsf{i}, k, j)$. Thus, $\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{does}(a^{\mathsf{i}}) \wedge \mathrm{WR}(\mathsf{i}, j) \wedge \mathsf{K}_{\mathsf{i}}(\bigwedge_{\mathsf{s}\in\mathrm{N}} \mathrm{Rat}(\mathsf{s}, k-1, j))$. Since R_{i} is reflexive, it follows that $\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{does}(a^{\mathsf{i}}) \wedge \mathrm{Rat}(\mathsf{i}, k-1, j)$.

□

Note that increasing j may not enable the elimination of actions. The larger j, the more stages will be considered. Ideally, j should be large enough to reach terminal states. However, termination may not be ensured in auction protocols and real world players usually have time restrictions to decide their actions.

4.2.2 Bounded Rationality in the Dutch Auction

Let us consider the Dutch auction from Section 4.1.3. Consider a specific instance $\mathrm{M}_{\mathrm{dut}}$ in $\mathcal{M}_{\mathrm{dut}}$, such that there are only two players i and s whose valuation for the good being auctioned is 7 and 4, respectively. The auctioneer starts by proposing the price 10 and in each round the price is decreased by 1. Formally, $\mathrm{N} = \{\mathsf{i}, \mathsf{s}\}, \mathrm{I} = \{0, \ldots, 10\}, \mathcal{B} = \{\mathrm{bid}_{\mathsf{i}}, \mathrm{wait}_{\mathsf{i}}, \mathrm{bid}_{\mathsf{s}}, \mathrm{wait}_{\mathsf{s}}\}, \Phi = \{\mathrm{winner}_{\mathsf{i}}, \mathrm{winner}_{\mathsf{s}}\}$ and $\mathrm{Y} = \{\mathrm{pay}_{\mathsf{i}}, \vartheta_{\mathsf{i}}, \mathrm{pay}_{\mathsf{s}}, \vartheta_{\mathsf{s}}\}$. Let $\mathrm{M}_{\mathrm{dut}}$ be the model defined by the signature $\mathcal{S}_{\mathrm{dut}} = (\mathrm{N}, \mathrm{I}, \mathcal{B}, \Phi, \mathrm{Y})$ and the constants $\mathsf{starting} = 10$, $\mathsf{dec} = 1$, $\mathsf{reserve} = 0$ and $\mathsf{n} = 2$. We consider the initial state $\mathrm{w}_0 \in \mathrm{W}_\iota$, such that $\pi_{\mathrm{Y}}(\mathrm{w}_0, \vartheta_{\mathsf{i}}) = 7$ and $\pi_{\mathrm{Y}}(\mathrm{w}_0, \vartheta_{\mathsf{s}}) = 4$.

Due to the starting price and the decrement, the auction is ensured to end after 10 stages. We therefore focus on the case $j = 10$. If the auction reaches a terminal state before 10 stages, the update function ensures a loop in the terminal state. Since the auction ends at the first bid, we write $\mathrm{bidAfter}(\mathsf{i}, m)$ as the sequence of actions σ^{i}, such that $\sigma^{\mathsf{i}}_i = \mathrm{wait}_{\mathsf{i}}$ for $i < m \leq j$ and $\sigma^{\mathsf{i}}_i = \mathrm{bid}_{\mathsf{i}}$ for $m \leq i \leq j$. The sequence is read "i bids after m steps". Let $\mathrm{onlywait}(\mathsf{i})$ be the sequence of j actions $\mathrm{wait}_{\mathsf{i}}$. We use a similar notation for expressing agent s's sequence of actions. Let d be a joint action, we will examine which sequences of actions are rational for each agent to follow. We assume the Dutch auction protocol Σ_{dut} and the tie-breaking ordering are common knowledge among the agents in N.

If the agents are 0-order rational, that is, if $\mathrm{M}_{\mathrm{dut}} \models_{(\mathrm{w}_0,d)} \mathrm{Rat}(\mathsf{i}, 0, j) \wedge \mathrm{Rat}(\mathsf{s}, 0, j)$, then both agents consider possible that any sequence of joint actions will be taken. If we now consider 1st-order rationality for i, that is $\mathrm{M}_{\mathrm{dut}} \models_{(\mathrm{w}_0,d)} \mathrm{Rat}(\mathsf{i}, 1, j)$, then i is not going to follow any sequence of actions that are strongly dominated in j steps. The weakly rational sequences of actions for i are those where she waits until the price is below her private valuation (e.g, $\mathrm{bidAfter}(\mathsf{i}, 4), \mathrm{bidAfter}(\mathsf{i}, 5)$, and so on). The sequence of actions $\mathrm{onlywait}(\mathsf{i})$ is not rational for i. The weakly rational actions for agent s when $\mathrm{M}_{\mathrm{dut}} \models_{(\mathrm{w}_0,d)} \mathrm{Rat}(\mathsf{s}, 1, j)$ are defined similarly. Figure 4.2 illustrates the utilities each agent considers possible to achieve when playing a weakly rational sequence of actions.

For $k > 1$, which actions a k-order rational agent considers possible her opponents will take depends on her knowledge about their valuations. For instance, let us consider the case where it is common knowledge that $(2 \leq \vartheta_{\mathsf{s}} \leq \mathsf{starting}) \wedge (2 \leq \vartheta_{\mathsf{i}} \leq$

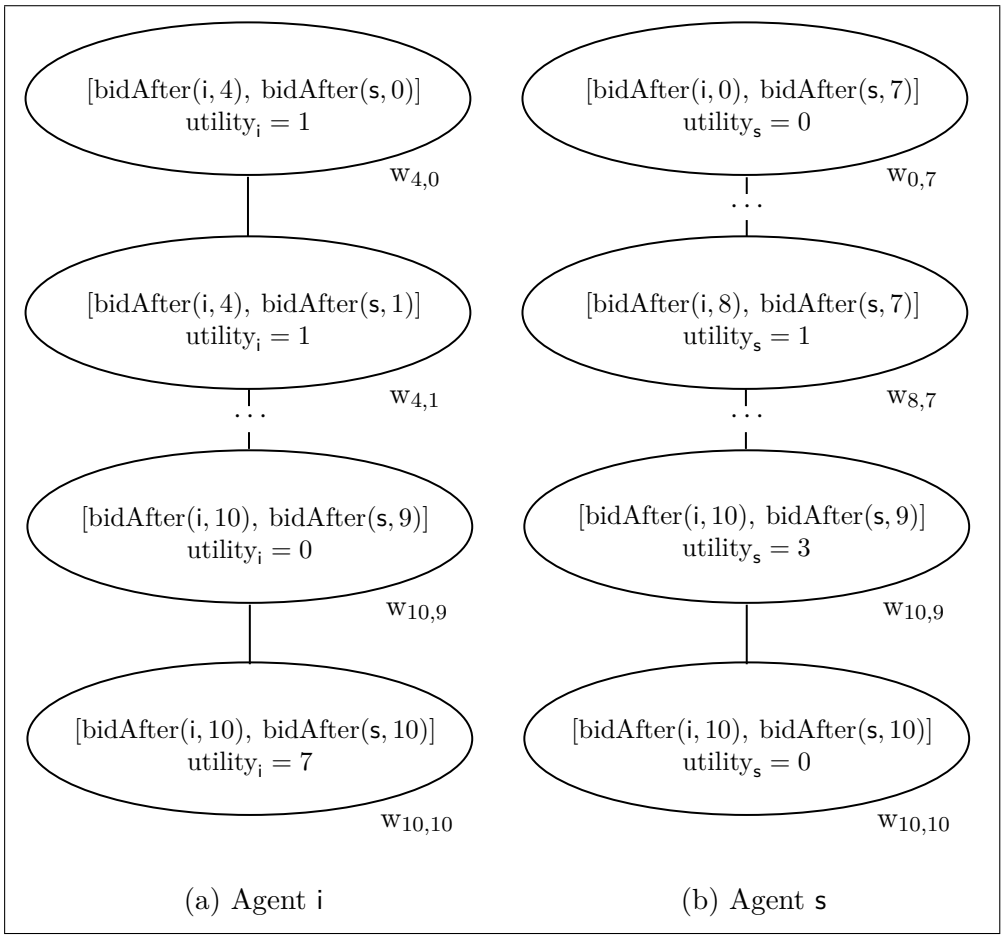

Figure 4.2: The utilities agents i and s consider possible to obtain when they are 1st-order rational

starting), *i.e.*, we have $M_{dut} \models (2 \leq \vartheta_s \leq \mathsf{starting}) \wedge (2 \leq \vartheta_i \leq \mathsf{starting})$. By Proposition 4.1, both agents then know their opponent has a valuation between 2 and the starting price. If the agent s is 2nd-order rational, she will know the sequence of actions onlywait(i) is not weakly rational for i. Due to the tie-breaking rule, if both agents bid at the same stage, agent i wins. Thus, agent s cannot win by waiting for the price to reach zero and it is not weakly rational to perform bidAfter(s, 10). If i is 3rd-order rational, she knows that s knows onlywait(i) is not rational for her and consequently, that it cannot be the case that s will bidAfter(s, 10). If the agents are 4th-order rational, they will not consider possible that the good is not sold before the price be zero. Thus, a similar reasoning will happen due tie-breaking when the price is 1. Finally, Figure 4.3 illustrates the utilities each agent considers possible when she is 7th-order rational. Since agents are uncertain about which value between 2 and starting represents the valuation of their opponents, raising the order of rationality beyond 7 would not modify the actions they consider possible to be taken by their opponent.

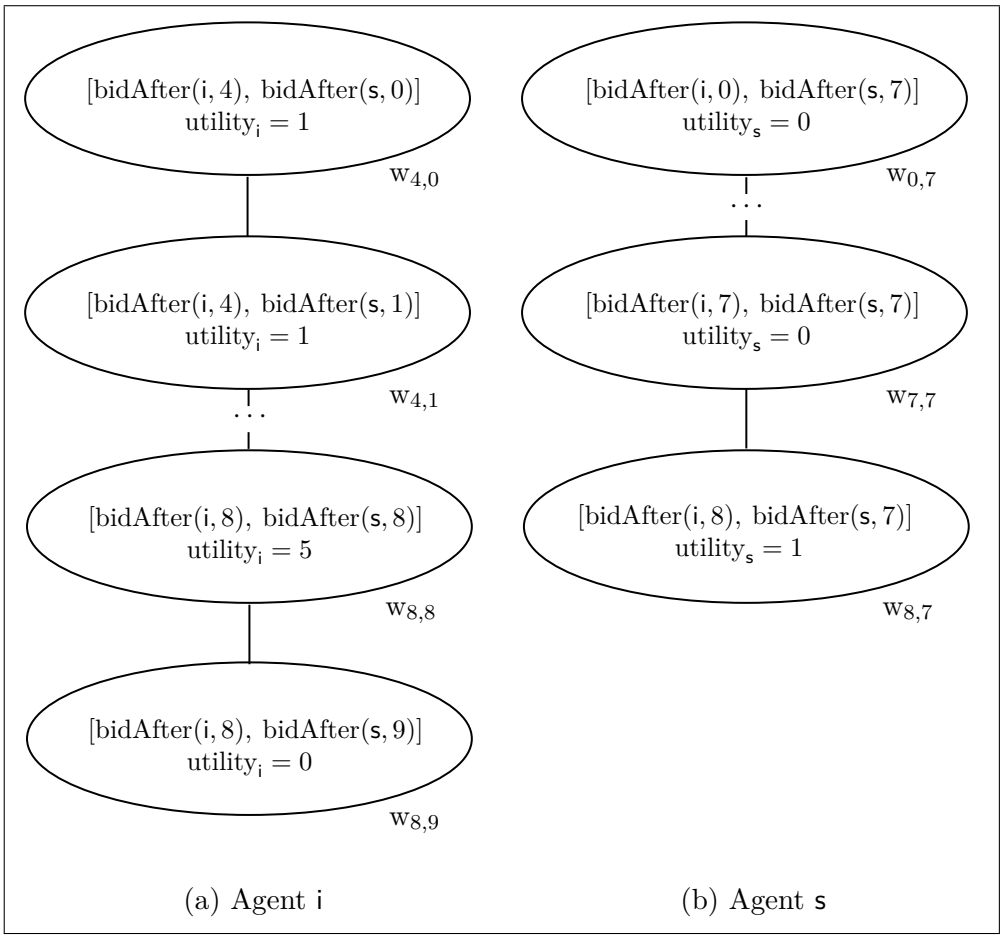

Figure 4.3: The utilities agents i and s consider possible to obtain when they are 7th-order rational and $\mathrm{M}_{\mathrm{dut}} \models (2 \leq \vartheta_{\mathsf{s}} \leq \mathsf{starting}) \wedge (2 \leq \vartheta_{\mathsf{i}} \leq \mathsf{starting})$

4.3 Model Checking

Now we examine the upper bound of the complexity of deciding whether an ADLKformula is true with respect to a model and a move. To prove this bound, we provide a model-checking algorithm and analyze its complexity. Let $\varphi \in \mathcal{L}_{\mathsf{ADLK}}$ be a formula and $\mathrm{M} = (\mathrm{W}, \mathrm{W}_\iota, \mathrm{T}, \{\mathrm{R}_{\mathsf{i}}\}_{\mathsf{i}\in\mathrm{N}}, \mathrm{U}, \pi_\Phi, \pi_\mathrm{Y})$ be an ET-Model over $\mathcal{S}$. We say that ψ is a subformula of φ if either (i) $\psi = \varphi$; (ii) φ is of the form $\neg\varphi$, $\mathsf{K}_{\mathsf{i}}\varphi$ or $[\,d^{\mathrm{C}}\,]\varphi$ and ψ is a subformula of φ; or (iii) φ is of the form $\varphi \wedge \varphi'$ and ψ is a subformula of either φ or φ'. Denote $Sub(\varphi)$ as the set of all subformulas of φ.

Theorem 4.2. *The following problem is in* $\mathcal{O}(|\mathrm{W}| \times |\mathcal{B}|^m)$*, where* $m = |\mathrm{N}| \times |\varphi|$*: Given an ET-Model* M*, a state* $\mathrm{w} \in \mathrm{W}$*, a joint action* $d \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^{\mathsf{i}}$ *and a formula* $\varphi \in \mathcal{L}_{\mathsf{ADLK}}$*, determine whether* $\mathrm{M} \models_{(\mathrm{w},d)} \varphi$ *or not.*

Proof. Algorithm 2, named *epistModelCheck*, works in the following way: first it gets all subformulas of φ and orders them in a vector S by ascending length. Thus,

Algorithm 2 $epistModelCheck(\mathrm{M}, \mathrm{w}, d, \varphi)$

Input: an ET-model $\mathrm{M} = (\mathrm{W}, \mathrm{W}_\iota, \mathrm{T}, \{\mathrm{R}_\mathsf{i}\}_{\mathsf{i}\in\mathrm{N}}, \mathrm{U}, \pi_\Phi, \pi_\Upsilon)$, a state w of W, a joint action $d \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^\mathsf{i}$ and a formula $\varphi \in \mathcal{L}_{\mathsf{ADLK}}$.

Output: **true** if $\mathrm{M} \models_{(\mathrm{w},d)} \varphi$, and **false** otherwise

1: $S \leftarrow Sub(\varphi)$ ordered by ascending length
2: Let $isTrue[1, \ldots, |\varphi|]$ be a boolean array initiated with *true* values
3: **for** $i \leftarrow 1$ to $|\varphi|$ **do**
4: $\quad\varphi \leftarrow S[i]$
5: $\quad$**switch** the formula type of φ **do**
6: $\quad\quad$**case** φ is of the form $\varphi' \wedge \varphi''$
7: $\quad\quad\quad isTrue[i] \leftarrow isTrue[getIndex(S, \varphi')] \wedge isTrue[getIndex(S, \varphi'')]$
8: $\quad\quad$**case** φ is of the form $\neg\varphi'$
9: $\quad\quad\quad isTrue[i] \leftarrow \neg isTrue[getIndex(S, \varphi')]$
10: $\quad\quad$**case** φ is atomic
11: $\quad\quad\quad isTrue[i] \leftarrow \mathrm{M} \models_{(\mathrm{w},d)} \varphi$
12: $\quad\quad$**case** φ is of the form $[\, b^\mathrm{C}\,]\varphi'$
13: $\quad\quad\quad e^\mathrm{C} \leftarrow \langle b^\mathrm{C}, d^{\text{-C}}\rangle$
14: $\quad\quad\quad$**for each** $c \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^\mathsf{i}$ **do**
15: $\quad\quad\quad\quad isTrue[i] \leftarrow isTrue[i] \wedge epistModelCheck(\mathrm{M}, \mathrm{U}(\mathrm{w}, e), c, \varphi')$
16: $\quad\quad$**case** φ is of the form $\mathsf{K}_\mathsf{i}\varphi'$
17: $\quad\quad\quad$**for each** $\mathrm{v} \in \mathrm{R}_\mathsf{i}(\mathrm{w})$ and each $e \in \prod_{\mathsf{i}\in\mathrm{N}} \mathrm{A}^\mathsf{i}$ with $e^\mathsf{i} = d^\mathsf{i}$ **do**
18: $\quad\quad\quad\quad isTrue[i] \leftarrow isTrue[i] \wedge epistModelCheck(\mathrm{M}, \mathrm{v}, e, \varphi')$
19: **return** $isTrue[|\varphi|]$

$S(|\varphi|) = \varphi$, *i.e.*, the position $|\varphi|$ in S corresponds to the formula φ itself, and if φ_i is a subformula of φ_j then $i < j$. An induction on S labels each subformula φ_i depending on whether or not φ_i is true in M at the move (w, d). If φ_i does not have any subformula, its truth value is obtained directly from the model. Since S is ordered by formulas length, if φ_i is either of the form $\varphi' \wedge \varphi''$ or $\neg\varphi'$ the algorithm labels φ_i according to the label assigned to φ' and/or φ''. If φ_i is of the form $[\, b^\mathrm{C}\,]\varphi'$ then its label is recursively defined according to φ' truth value in the updated state given the joint action $\langle b^\mathrm{C}, d^{\text{-C}}\rangle$, for any joint action to be taken in the next move. Since we compare with every joint action, this is done in an exponential number of steps, based on the size of the set of agents (*i.e.*, according to $|\mathcal{B}|^n$, where $n = |\mathrm{N}|$). Finally, the case where φ_i is in the form $\mathsf{K}_\mathsf{i}\varphi'$ is recursively defined according to the truth value of φ' in all moves that are equivalent to (w, d). Similar to the previous case, since we compare with all possible moves and all states in $\mathrm{R}_\mathsf{i}(\mathrm{w}) \subseteq \mathrm{W}$, this step is done in an exponential number of steps (*i.e.*, according to $|\mathrm{W}| \times |\mathcal{B}|^n$, where $n = |\mathrm{N}|$). As Algorithm *epistModelCheck* visits each subformula at most once, and the number of subformulas is not greater than the size of φ, the algorithm can clearly be implemented in $\mathcal{O}(|\mathrm{W}| \times |\mathcal{B}|^m)$, where $m = |\mathrm{N}| \times |\varphi|$. □

It follows that checking agent rationality is exponential in the quantity of agents,

the order of rationality and how many rounds are considered.

Corollary 4.1. *Given an ET-model* M*, a state* $\mathrm{w} \in \mathrm{W}$*, a joint action* $d \in \prod_{\mathsf{i} \in \mathrm{N}} \mathrm{A}^{\mathsf{i}}$*, an agent* i*,* $j > 0$ *and* $k > 0$*, the problem of checking whether* $\mathrm{M} \models_{(\mathrm{w},d)} \mathrm{Rat}(\mathsf{i}, k{+}1, j)$ *is in* $\mathcal{O}(|\mathrm{W}| \times |\mathcal{B}|^{nkj})$*, where* $n = |\mathrm{N}|$*.*

4.4 Conclusion

In this chapter, we presented Epistemic Auction Description Language (ADLK), a language to allow reasoning about knowledge, rationality and action choice in auctions. ADLK extend ADL with epistemic operators and action modalities as in the GGP competition, real world bidders may have time restrictions to decide their actions. Thus, we characterized and explored bounded rationality in relation to the level of higher-order knowledge about other agents and stages to *look-ahead* beyond the current state.

With ADLK, we can reason about agents' action choices over a finite number of auction stages, but it does not contemplate agents' strategies in a general sense (that is, functions describing what to do in each possible moment of a game). In the second part of this thesis, we investigate strategic reasoning in auctions and explore how to create and verify mechanisms which will be played by strategic agents. We provide a new perspective for Automated Mechanism Design, grounded on formal methods and logic-based automated reasoning.

Part II

Strategic Reasoning in Mechanism Design

Chapter 5

Verification of Mechanisms

Let us recall the problem of Mechanism Design, which consists on creating games that aggregate agents' preferences towards a single joint decision. When participants act rationally (in the game theoretical sense), such games should ensure a preferable behavior of the players as well as desirable features of the decision (Nisan et al., 2007).

Bearing in mind that the mechanism analysis is usually a manual process, we investigate their verification in relation to complex properties involving strategic agents through formal methods. In this context, the paper "Logic for Mechanism Design - A Manifesto" (Pauly and Wooldridge, 2003) is of particular interest. The authors argue that strategic logics developed for the formal verification of multi-agent systems (MAS) could be good candidates as formal frameworks to reason about mechanisms. They consider Alternating-time Temporal Logic (ATL) (Alur et al., 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 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." (Pauly and Wooldridge, 2003)

In this chapter we argue that Strategy Logic (Chatterjee et al., 2010) is a good candidate for a general-purpose logic for mechanism design, specially when considering auction-based markets. More precisely, we propose a new variant of the logic with quantitative features, imperfect information and epistemic operators, that we call $\mathsf{SLK}[\mathcal{F}]$. 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 start by presenting the related work on automated mechanism design and logics for strategic reasoning. Next, we propose $\mathsf{SLK}[\mathcal{F}]$ and show how it can be used for

reasoning about auction mechanisms. Finally, we investigate the model-checking problem for this language and conclude the chapter

5.1 Related Work

In this section we present the related work on computer-aided and fully-automatic approaches for mechanism design and we recall logic-based languages for reasoning about strategic abilities in MAS.

5.1.1 Automated Mechanism Design

Traditionally, mechanisms have been formulated by specialists, who use their knowledge and experience for defining the game rules. Conitzer and Sandholm (2003) 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, neural networks have been used to learn mechanisms that optimize a given parameter, such as revenue (Shen et al., 2019; Dütting et al., 2019). Statistical machine learning techniques have also been considered in domains without money (Narasimhan et al., 2016). Vorobeychik et al. (2007) proposes a black-box optimization algorithm for evaluating candidate mechanisms. Evolutionary search methods have also been used by Niu et al. (2012) to optimize double auctions, while Asselin et al. (2006) addresses AMD through linear programming and optimization. By treating AMD as an engineering problem, Phelps et al. (2010) focus on evolutionary and iterative approaches to mechanism design.

In relation to automating the analysis of mechanisms, some works from computer-aided verification (Caminati et al., 2015; Barthe et al., 2016; Kerber et al., 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 strategyproofness 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.

A related topic is normative systems Ågotnes et al. (2007), which defines constraints (in terms of *obligations* and *permissions*) on the behaviour of agents. Bulling and Dastani (2016) investigates how concepts from mechanism design can be used to analyse the enforcement of norms with preferences modelled using Linear-time Temporal Logic (LTL).

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 ATL (Alur et al., 2002). They discuss that ATL lacks the ability to reason about quantitative aspects such

as preferences, and game-theoretic concepts such as equilibria. Okada et al. (2019) uses Boolean Satisfiability for modeling mechanisms for false-name-proof facility location. Their approach is restricted to one type of mechanism and does not handle strategic, temporal and quantitative specifications. Another related approach are Boolean games (Harrenstein et al., 2001), which aims at providing models that are easy to build with actions determined by the agents' control of propositional variables. Finally, Gutierrez et al. (2019) considers system specifications given in LTL (Vardi, 1996), and studies the implementation of a mechanism to ensure a given temporal logic property in equilibrium.

5.1.2 Logics for Strategic Reasoning

Our work is rooted in a rich line of work on logics for strategic reasoning, starting with the aforementioned ATL (Alur et al., 2002), the foundational language for strategic reasoning in MAS. 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). The first order extension of ATL (Belardinelli and Lomuscio, 2016) allows one to capture some quantitative aspects. 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. Strategy Logic (SL) (Chatterjee et al., 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, hierarchical 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. We denote by Epistemic Strategy Logic (SLK) the logic resulted of augmenting SL with knowledge operators Maubert and Murano (2018). Recently, SL[$\mathcal{F}$] (Bouyer et al., 2019) 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 chapter we merge both lines by combining quantitative aspects and imperfect information in SLK[$\mathcal{F}$].

Indeed SL[$\mathcal{F}$], which subsumes ATL, is expressive enough to express complex solution concepts such as Nash equilibrium and properties about quantities. This language thus allows for specifications that contain constraints on mechanism properties (for instance, in Auction Design, the efficiency and budget-balance). Its quantitative semantics, with satisfaction values that reflect how well a model satisfies a formula, also allows us to investigate the constructions of mechanisms that approximate such properties, which is not possible with standard SL.

A key assumption of the present contribution is that agents have only partial observability of the global state of the system, as it is often the case in real-life applications. Contexts of imperfect information have been extensively considered

in the literature on formal verification (Dima and Tiplea, 2011; Kupferman and Vardi, 2000; Jamroga and Ågotnes, 2007; Reif, 1984; Bulling and Jamroga, 2014). Generally speaking, imperfect information immediately entails higher complexity of game solving. In multi-player games, the complexity can go up to being non-elementary (Pnueli and Rosner, 1989), or even undecidability when considered in the context of memoryful strategies (Dima and Tiplea, 2011). Hence, it is of interest to analyse imperfect information systems where agents have finite or bounded memory, in order to retrieve a decidable model checking problem. Herzig et al. (2016) analyse epistemic boolean games in a computationally grounded dynamic epistemic logic with the model-checking in in PSPACE.

In relation to the modeling, specification, and reasoning about strategies of bounded-memory agents, Ågotnes and Walther (2009) investigate strategic abilities of agents with bounded memory, while Belardinelli et al. (2018) consider bounded memory as an approximation of perfect recall. On a related direction, temporal and strategic logics have been extended to handle agents with bounded resources (Alechina et al., 2009, 2010; Bulling and Farwer, 2010a,b). Issues related to bounded rationality are also investigated in (Barlo et al., 2008; Hörner and Olszewski, 2009; Gupta et al., 2015).

Also relevant for the present contribution are papers that study explicit representations of strategies. This category is much richer and includes extensions of ATL^* with explicit reasoning about actions and strategies (van der Hoek et al., 2005; Ågotnes, 2006; Walther et al., 2007; Herzig et al., 2014), as well as logics that combine features of temporal and dynamic logic (Harel and Kozen, 1982; Novák and Jamroga, 2009). Duijf and Broersen (2016) present a variant of STIT logic[1], that enables reasoning about strategies and their execution in the object language. Also, plans in agent-oriented programming are in fact rule-based descriptions of strategies. In particular, reasoning about agent programs using strategic logics was investigated in (Bordini et al., 2006; Alechina et al., 2007, 2008; Dastani and Jamroga, 2010; Yadav and Sardiña, 2012).

The work in (Jamroga et al., 2019a,b) proposed to model "human-friendly" strategies by lists of condition-action pairs with bounded complexity. This was in contrast to "combinatorial" strategies, defined as functions from (sequences of) states to actions, and typically used in the semantics of MAS logics (Alur et al., 2002; Pauly and Parikh, 2003; Chatterjee et al., 2010; Mogavero et al., 2014). It was argued in (Jamroga et al., 2019a,b) that natural strategies provide better models of behavior for agents with limited memory and computing capacity, such as humans or simple bots. The concept have been already used to redefine some security requirements for voting protocols in (Jamroga et al., 2020).

[1] Also known as the logic of *"Seeing to it That"*, first proposed by Belnap and Perloff (1990).

5.2 Quantitative Epistemic Strategy Logic

$\mathsf{SL}[\mathcal{F}]$ (Bouyer et al., 2019) introduces quantitative aspects in SL, but it lacks the ability to handle imperfect information inherent to the mechanism 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 this thesis, we fix a set of atomic propositions AP, a set of agents N and a set of strategy variables Var, except when stated otherwise. We let n be the number of agents in N. Finally, 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 that will parameterise the logics we consider.

Definition 5.1. The syntax of $\mathsf{SLK}[\mathcal{F}]$ is defined by the following grammar:

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

where $p \in \mathrm{AP}$, $s_{\mathsf{i}} \in \mathrm{Var}$, $\mathsf{i} \in \mathrm{N}$, and $f \in \mathcal{F}$.

The intuitive reading of the operators is as follows: $\exists s_{\mathsf{i}}.\, \varphi$ means that there exists a strategy for agent i such that φ holds; $(\mathsf{i}, s_{\mathsf{i}})\varphi$ means that when strategy s_{i} is assigned to agent i, φ holds; $K_{\mathsf{i}}\varphi$ means that agent i knows that φ holds; $\mathbf{X}$ and $\mathbf{U}$ are the usual temporal operators "next" and "until". The meaning of $f(\varphi_1, \dots, \varphi_n)$ 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 5.1. In (Bouyer et al., 2019), 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 i 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 i. The set of free variables and agents in φ is written $\mathrm{free}(\varphi)$, and a formula φ is a *sentence* if $\mathrm{free}(\varphi) = \emptyset$.

The strategy quantifier $\exists s_{\mathsf{i}}.\, \varphi$ quantifies on strategies *for agent* i. Except in its original formulation (Chatterjee et al., 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 5.2. A *weighted concurrent game structure with imperfect information* (wCGSii) is a tuple $\mathcal{G} = (\{\mathcal{B}_\mathsf{i}\}_{\mathsf{i}\in\mathrm{N}}, V, \delta, \ell, V_\iota, \{\sim_\mathsf{i}\}_{\mathsf{i}\in\mathrm{N}})$ where

- $\mathcal{B}_\mathsf{i}$ is a finite set of *actions* for agent i;
- V is a finite set of *positions*;
- $\delta : V \times (\prod_{\mathsf{i}\in\mathrm{N}} \mathcal{B}_\mathsf{i}) \to V$ 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*;
- $\sim_\mathsf{i} \subseteq V \times V$ is an equivalence relation called the *observation relation* of agent i.

For a collection of objects indexed by agents in N, we may omit the index set and write, e.g., $\{\sim_\mathsf{i}\}$ for $\{\sim_\mathsf{i}\}_{\mathsf{i}\in\mathrm{N}}$. We will also often 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 $\mathrm{N}_{-\mathsf{i}} = \mathrm{N} \setminus \{\mathsf{i}\}$. This notation is also used in the subsequent chapters.

Action profiles In a position $v \in V$, each player i chooses an action $a_\mathsf{i} \in \mathcal{B}_\mathsf{i}$, and the game proceeds to position $\delta(v, \boldsymbol{a})$ where $\boldsymbol{a}$ is the *action profile* $(a_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$.

Plays A *play* $\pi = v_0 v_1 v_2 \ldots$ is an infinite sequence of positions such that for every $i \geq 0$ 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.

Strategies A (memoryless) *strategy* for agent i is a function $\sigma : V \to \mathcal{B}_\mathsf{i}$ that maps each position to an action. A strategy σ for agent i is *uniform* if, for all positions v, v' such that $v \sim_\mathsf{i} v'$, we have $\sigma(v) = \sigma(v')$. We let Str_i be the set of uniform strategies for agent i, and $Str = \cup_{\mathsf{i}\in\mathrm{N}} Str_\mathsf{i}$. Because there are finitely many positions, Str is finite.

Assignments An *assignment* $\mathcal{A} : \mathrm{N} \cup \mathrm{Var} \to Str$ is a function from players and variables to strategies. For an assignment $\mathcal{A}$, an agent i and a strategy σ for i, $\mathcal{A}[\mathsf{i} \mapsto \sigma]$ is the assignment that maps i 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 $v_0 v_1 \dots$ such that $v_0 = v$ and for all $i \geq 0$, $v_{i+1} = \delta(v_i, \boldsymbol{a})$ where for all $\mathsf{i} \in \mathrm{N}$, $\boldsymbol{a}_\mathsf{i} = \mathcal{A}(\mathsf{i})(v_0 \dots v_i)$.

Definition 5.3. Let $\mathcal{G} = (\{\mathcal{B}_\mathsf{i}\}_{\mathsf{i}\in\mathrm{N}}, V, \delta, \ell, V_\iota, \{\sim_\mathsf{i}\}_{\mathsf{i}\in\mathrm{N}})$ be a wCGSii, and $\mathcal{A}$ an assignment. The satisfaction value $[\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) \in [-1, 1]$ of an $\mathsf{SLK}[\mathcal{F}]$ formula φ in a position v is defined as follows, where π denotes $\mathrm{Out}(v, \mathcal{A})$:

$$
\begin{aligned}
[\![p]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \ell(v, p) \\
[\![\exists s_\mathsf{i}.\, \varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \max_{\sigma \in Str_\mathsf{i}} [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[s_\mathsf{i} \mapsto \sigma]}(v) \\
[\![(\mathsf{i}, s_\mathsf{i})\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[\mathsf{i} \mapsto \mathcal{A}(s_\mathsf{i})]}(v) \\
[\![K_\mathsf{i}\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \min_{v' \sim_\mathsf{i} v} [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v') \\
[\![f(\varphi_1, \dots, \varphi_m)]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= f([\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(v), \dots, [\![\varphi_m]\!]^{\mathcal{G}}_{\mathcal{A}}(v)) \\
[\![\mathbf{X}\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_1) \\
[\![\varphi_1 \mathbf{U} \varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \sup_{i \geq 0} \min\Big([\![\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_i), \min_{0 \leq j < i} [\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_j)\Big)
\end{aligned}
$$

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: $\bot =_{\mathrm{def}} \neg\top$, $\varphi \wedge \varphi' =_{\mathrm{def}} \neg(\neg\varphi \vee \neg\varphi')$, $\varphi \to \varphi' =_{\mathrm{def}} \neg\varphi \vee \varphi'$, $\mathbf{F}\psi =_{\mathrm{def}} \top\mathbf{U}\psi$, $\mathbf{G}\psi =_{\mathrm{def}} \neg\mathbf{F}\neg\psi$ and $\forall s.\, \varphi =_{\mathrm{def}} \neg\exists s.\, \neg\varphi$, and check that they correspond to the intuition. For instance, $\wedge$ corresponds to min, $\mathbf{F}\psi$ computes the supremum of the satisfaction value of ψ over all future points in time, $\mathbf{G}\psi$ computes the infimum of these values[2], and $\forall s.\, \varphi$ minimizes the value of φ over all possible strategies s.

Remark 5.2. 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.

5.3 Reasoning about Auction Mechanisms

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

5.3.1 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.

[2]The supremum of a set is its least upper bound and the infimum is its greatest upper bound.

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 $\alpha = (x, \mathrm{p})$ where $x \in \mathcal{X}$ is a *choice* from a finite set of choices $\mathcal{X}$, and $\mathrm{p}_\mathsf{i} \in [-1, 1]$ is a payment for agent i.

For each agent $\mathsf{i} \in \mathrm{N}$, let also $\Theta_\mathsf{i} \subset [-1, 1]$ be a finite set of possible *types* for i. The type of an agent determines her preferences over the alternatives via the utility function introduced later on. Each type is an abstraction of the agents' preference and not its explicit representation[3]. We let $\boldsymbol{\Theta} = \prod_{\mathsf{i}\in\mathrm{N}} \Theta_\mathsf{i}$, and we note $\boldsymbol{\theta} = (\theta_\mathsf{i})_{\mathsf{i}\in\mathrm{N}} \in \boldsymbol{\Theta}$ 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* $\mathsf{v}_\mathsf{i} : \mathcal{X} \times \Theta_\mathsf{i} \to [-1, 1]$.

Example 5.1. For instance, in a one-sided auction with one good, a choice describes who wins the good. This can be modelled by letting $\mathcal{X} = \{-1, 1\}^\mathrm{N}$, and considering that choice $(x_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$ represents that agent i wins the good if $x_\mathsf{i} = 1$. 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 $\mathcal{X} = [-1, 1]^\mathrm{N}$, and a choice $(x_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$ means that agent i buys x_i items if $x_\mathsf{i} \geq 0$, and sells $-x_\mathsf{i}$ items if $x_\mathsf{i} \leq 0$ (values are normalized in [-1,1]).

The type θ_i of agent i reflects how much the agent desires the good. In one-sided auctions with choices defined as $\mathcal{X} = \{-1, 1\}^\mathrm{N}$ as in Example 5.1, one could define the valuation of agent i as $\mathsf{v}_\mathsf{i}(x, \theta_\mathsf{i}) = x_\mathsf{i} \cdot \theta_\mathsf{i}$. With this definition, a type θ_i 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 $\mathcal{X} = [-1, 1]^\mathrm{N}$, a positive type indicates the quantity an agent is willing to buy, while a negative type denotes a selling quantity. When defining the valuation of agent i as $\mathsf{v}_\mathsf{i}(x, \theta_\mathsf{i}) = x_\mathsf{i} \cdot \theta_\mathsf{i}$, 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 i with type θ_i for an alternative $\alpha = (x, \mathrm{p})$ is defined as

$$u_\mathsf{i}(\alpha, \theta_\mathsf{i}) = \mathsf{v}_\mathsf{i}(x, \theta_\mathsf{i}) - \mathrm{p}_\mathsf{i}$$

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

Definition 5.4. A *social choice function* (SCF) $\mathsf{s} : \boldsymbol{\Theta} \to \mathrm{Alt}$ is a function that, given a type profile $\boldsymbol{\theta}$, chooses an alternative $\mathsf{s}(\boldsymbol{\theta}) \in \mathrm{Alt}$. We can split a social

[3]The meaning of a type depends on the problem being considered and it should not be necessarily understood in the numerical/ordinal sense. We assume the sets of possible types are finite, as done in the literature (Sandholm, 2003).

choice function as follows: $\mathsf{s} = (\mathsf{x}, \{\mathsf{p_i}\})$, where $\mathsf{x} : \boldsymbol{\Theta} \to \mathcal{X}$ is a *choice function* and for each i, $\mathsf{p_i} : \boldsymbol{\Theta} \to [-1, 1]$ is a *payment function* for agent i.

Example 5.2. The first-price social choice function $\mathsf{s_{fp}} = (\mathsf{x}, \{\mathsf{p_i}\})$ is defined as follows. The allocation choice is defined as $\mathsf{x}(\boldsymbol{\theta}) = (x_1, \ldots, x_\mathsf{n})$, where $x_\mathsf{i} = 1$ if θ_i is the highest type in $\boldsymbol{\theta}$ and $x_\mathsf{i} = 0$ otherwise. In case two agents $\mathsf{i} \neq \mathsf{r}$ have the highest type, $x_\mathsf{i} = 1$ iff $\mathsf{i} \prec \mathsf{r}$. The payment function for agent i is defined as $\mathsf{p_i}(\boldsymbol{\theta}) = x_\mathsf{i} \cdot \theta_\mathsf{i}$.

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

5.3.2 Mechanisms as wCGSii

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 $\mathrm{G} = \{1, \ldots, \mathsf{m}\}$. An *allocative choice* (Parkes and Ungar, 2001) is a tuple $(x_\mathsf{i})_{\mathsf{i} \in \mathrm{N}} \in \mathcal{X}$ where $x_\mathsf{i} = (x_{\mathsf{i},1}, \ldots, x_{\mathsf{i},\mathsf{m}})$ denotes the allocation for agent i, and $x_{\mathsf{i},j} \in [-1, 1]$ is the amount of goods of type j allocated to agent i (normalized in $[-1, 1]$). Note that, in the case of one type of goods ($\mathsf{m} = 1$), we obtain $\mathcal{X} = [-1, 1]^\mathrm{N}$ as in Example 5.1.

Definition 5.5. Let $\mathrm{AP} \supseteq \{\mathrm{all}_{\mathsf{i},j}, \mathrm{pay}_\mathsf{i}, \mathrm{terminal}, \mathrm{type}_\mathsf{i} : \mathsf{i} \in \mathrm{N}, j \in \mathrm{G}\}$ be a set of atomic propositions, where $\mathrm{all}_{\mathsf{i},j}$, type_i, pay_i denote, respectively, how many units of the good j are allocated to agent i, the type of i, and her payment. The proposition terminal specifies whether a state is terminal. A *mechanism* is a wCGSii 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 $\delta(v, \boldsymbol{a}) = v'$ then $\ell(v, \mathrm{type}_\mathsf{i}) = \ell(v', \mathrm{type}_\mathsf{i})$ for each i;

(iii) each agent knows her own type: if $v \sim_\mathsf{i} v'$, then $\ell(v, \mathrm{type}_\mathsf{i}) = \ell(v', \mathrm{type}_\mathsf{i})$;

(iv) every play eventually reaches a *terminal position*, i.e., a sink[4] where proposition terminal has value 1;

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

(v) in all non-terminal positions, terminal 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 $\mathrm{all}_{\mathsf{i},j}$ and pay_i in terminal position $v(\boldsymbol{\theta}, \boldsymbol{\sigma})$ encode an alternative that we write $\mathcal{G}[\boldsymbol{\theta}, \boldsymbol{\sigma}]$.

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

Example 5.3 (Dutch auction). A Dutch auction is an iterative protocol with decreasing price; hereafter we assume the single good and single unit case. As introduced in Chapter 4, the auctioneer initially 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 $\mathsf{i} \in \mathrm{N}$, (i) a finite set of possible types $\Theta_\mathsf{i} \subset [0, 1]$, and (ii) her real type $\theta_\mathsf{i} \in \Theta_\mathsf{i}$. Agent i's valuation is $\mathsf{v}_\mathsf{i}(x) = x_\mathsf{i} \cdot \theta_\mathsf{i}$.

Define the mechanism $\mathcal{G}_{\mathrm{dut}} = (\{\mathcal{B}_\mathsf{i}\}, V, \delta, \ell, V_\iota, \{\sim_\mathsf{i}\})$ over $\mathrm{AP} = \{\mathrm{price}, \mathrm{all}_\mathsf{i}, \mathrm{pay}_\mathsf{i}, \mathrm{terminal}, \mathrm{type}_\mathsf{i} : \mathsf{i} \in \mathrm{N}\}$, where:

- $\mathcal{B}_\mathsf{i} = \{\mathrm{bid}, \mathrm{wait}\}$ for each $\mathsf{i} \in \mathrm{N}$,
- V consists of positions of the form $\langle \mathrm{pr}, \{x_\mathsf{i}\}, \mathrm{ter}, \{\theta_\mathsf{i}\}\rangle$ with $\mathrm{pr} \in \{1 - x \cdot \mathsf{dec} : 0 \leq x \leq \frac{1}{\mathsf{dec}}\}$ denoting the current price, $\mathrm{ter} \in \{-1, 1\}$ denoting whether the position is terminal, $x_\mathsf{i} \in \{0, 1\}$ specifying the allocation for agent i, and $\theta_\mathsf{i} \in \Theta_\mathsf{i}$ 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 $V_\iota = \{\langle 1, 0, \ldots, 0, 0, \theta_1, \ldots, \theta_\mathsf{n}\rangle \in V\}$.

In non-terminal states, the transition function keeps decreasing the price pr as long as it is above zero and every agent performs the action of waiting. If an agent i bids, the good is assigned to her ($x_\mathsf{i} = 1$). 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 ($\mathrm{ter} = 1$). The transition function defines a loop for terminal states, ensuring no change occurs in the auction afterwards (see Figure 5.1 for a partial illustration). Formally, for each position $v = \langle \mathrm{pr}, \{x_\mathsf{i}\}, \mathrm{ter}, \{\theta_\mathsf{i}\}\rangle$ and joint action $\boldsymbol{a} = (a_\mathsf{i})_{\mathsf{i} \in \mathrm{N}}$, transition $\delta(v, \boldsymbol{a})$ is defined as follows:

- If $\text{ter} = -1$, $\delta(v, \boldsymbol{a}) = \langle \text{pr}', \{x'_{\mathsf{i}}\}, \text{ter}', \{\theta_{\mathsf{i}}\}\rangle$ where:

$$\text{pr}' = \begin{cases} \text{pr} - \mathsf{dec} & \text{if pr} - \mathsf{dec} \geq 0 \text{ and} \\ & a_{\mathsf{i}} = \text{wait for all } \mathsf{i} \in \text{N} \\ \text{pr} & \text{otherwise} \end{cases}$$

$$x'_{\mathsf{i}} = \begin{cases} 1 & \text{if } a_{\mathsf{i}} = \text{bid and for all } \mathsf{i}' \neq \mathsf{i} \\ & \text{either } a_{\mathsf{i}'} = \text{wait or } \mathsf{i} \prec \mathsf{i}' \\ 0 & \text{otherwise} \end{cases}$$

$$\text{ter}' = \begin{cases} 1 & \text{if pr}' = \text{pr}, \\ -1 & \text{otherwise} \end{cases}$$

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

For each $v = \langle \text{pr}, \{x_{\mathsf{i}}\}, \text{ter}, \{\theta_{\mathsf{i}}\}\rangle$ and each $\mathsf{i} \in \text{N}$, the weight function is defined as follows: $\ell(v, \text{price}) = \text{pr}$, $\ell(v, \text{all}_{\mathsf{i}}) = x_{\mathsf{i}}$, $\ell(v, \text{pay}_{\mathsf{i}}) = x_{\mathsf{i}} \cdot \text{pr}$, $\ell(v, \text{terminal}) = \text{ter}$, and $\ell(v, \text{type}_{\mathsf{i}}) = \theta_{\mathsf{i}}$.

Finally, for each agent $\mathsf{i} \in \text{N}$ and for any two positions $v = \langle \text{pr}, \{x_{\mathsf{i}}\}, \text{ter}, \{\theta_{\mathsf{i}}\}\rangle$ and $v' = \langle \text{pr}', \{x'_{\mathsf{i}}\}, \text{ter}', \{\theta'_{\mathsf{i}}\}_{\mathsf{i}\in\text{N}}\rangle$ in V, the observation relation $\sim_{\mathsf{i}}$ is defined as follows: $v \sim_{\mathsf{i}} v'$ if (i) $\text{pr} = \text{pr}'$; (ii) $x_{\mathsf{r}} = x'_{\mathsf{r}}$, for all $\mathsf{r} \in \text{N}$; (iii) $\theta_{\mathsf{i}} = \theta'_{\mathsf{i}}$; and (iv) $\text{ter} = \text{ter}'$.

Observation relations $\sim_{\mathsf{i}}$ 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.

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

5.3.3 Implementation of Social Choice Functions

For analysing a mechanism, we use concepts of Game Theory, such as the implementation of social choice functions. The implementation of a SCF says whether the alternative the equilibrium solution of the mechanism corresponds to the alternative chosen by the SCF. In order to define it formally, we first introduce basic concepts and 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

$$\text{util}_{\mathsf{i}} =_{\text{def}} \mathsf{v}_{\mathsf{i}}((\text{all}_{1,1}, \ldots, \text{all}_{\mathsf{n},\mathsf{m}}), \text{type}_{\mathsf{i}}) - \text{pay}_{\mathsf{i}}$$

in the terminal situation.

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

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

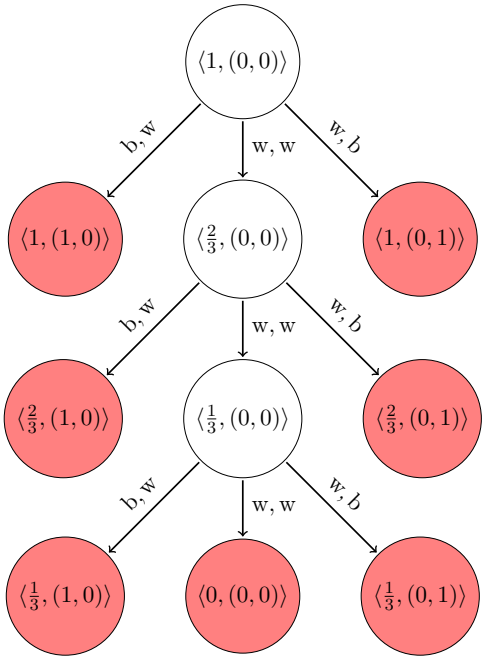

Figure 5.1: Part of the mechanism for the Dutch auction with two agents and decrement $\mathsf{dec} = \frac{1}{3}$. Terminal states are in red. We only represent one initial state and thus we omit types, which are the same in all states. Action bid is written b and wait is w. Finally, we did not represent ties (bid, bid) or loops.

as well as the valuation function $\mathsf{v_i}$ for each agent i, 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

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

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

We recall two classical concepts of equilibria, Nash equilibria and dominant strategy equilibria, classically used to define implementation.

Nash Equilibria A strategy profile $\sigma = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ 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

$$\mathrm{NE}(\boldsymbol{s}) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N}} \forall t.\, [(\mathrm{N}_{-\mathsf{i}}, \boldsymbol{s}_{-\mathsf{i}})(\mathsf{i}, t)\mathbf{F}(\mathrm{terminal} \wedge \mathrm{util}_{\mathsf{i}}) \leq (\mathrm{N}, \boldsymbol{s})\mathbf{F}(\mathrm{terminal} \wedge \mathrm{util}_{\mathsf{i}})]$$

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

Lemma 5.1. *For every assignment $\mathcal{A}$, we have that $[\![\mathrm{NE}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 1$ iff $(\mathcal{A}(s_{\mathsf{i}}))_{\mathsf{i}\in\mathrm{N}}$ is a NE in $\mathcal{G}$ from v.*

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

$$\begin{aligned}\mathrm{DS}(s_{\mathsf{i}}, \mathsf{i}) =_{\mathrm{def}} \forall \boldsymbol{t}.\big[&(\mathsf{i}, t_{\mathsf{i}})(\mathrm{N}_{-\mathsf{i}}, \boldsymbol{t}_{-\mathsf{i}})\mathbf{F}(\mathrm{terminal} \wedge \mathrm{util}_{\mathsf{i}}) \\ &\leq (\mathsf{i}, s_{\mathsf{i}})(\mathrm{N}_{-\mathsf{i}}, \boldsymbol{t}_{-\mathsf{i}})\mathbf{F}(\mathrm{terminal} \wedge \mathrm{util}_{\mathsf{i}})\big]\end{aligned}$$

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

A strategy profile $\sigma = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ is a *dominant strategy equilibrium* (DSE) if each σ_{i} is a dominant strategy for agent i (Nisan et al., 2007). Define

$$\mathrm{DSE}(\boldsymbol{s}) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N}} \mathrm{DS}(s_{\mathsf{i}}, \mathsf{i})$$

Similarly to Nash equilibria, the following holds.

Lemma 5.2. *For every assignment $\mathcal{A}$, we have that $[\![\mathrm{DSE}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 1$ iff $(\mathcal{A}(s_{\mathsf{i}}))_{\mathsf{i}\in\mathrm{N}}$ is a* DSE *in $\mathcal{G}$ from v.*

Remark 5.3. 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. Hereafter, we focus on these two concepts.

Definition 5.6. Let $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$ be a solution concept and s a social choice function. A mechanism $\mathcal{G}$ E-*implements* s if for all type profiles $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ there exists an E-equilibrium $\boldsymbol{\sigma}$ in $\mathcal{G}$ from $v_\iota^{\boldsymbol{\theta}}$ such that $\mathcal{G}[\boldsymbol{\theta}, \boldsymbol{\sigma}] = \mathsf{s}(\boldsymbol{\theta})$.

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

Example 5.4. Under the assumption that a Nash equilibrium exists, the Dutch auction (see Example 5.3) is known to implement the first-price social choice function (Krishna, 2009) introduced in Example 5.2.

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

$$\varphi_{\mathsf{s}(\boldsymbol{\theta})} =_{\mathrm{def}} \bigwedge_{\mathsf{i} \in \mathrm{N}} (\mathrm{pay}_\mathsf{i} = \mathrm{p}_\mathsf{i} \wedge \bigwedge_{j \in \mathrm{G}} \mathrm{all}_{\mathsf{i},j} = x_{\mathsf{i},j})$$

where $\mathsf{x}(\boldsymbol{\theta}) = ((x_{\mathsf{i},1}, \ldots, x_{\mathsf{i},\mathsf{m}}))_{\mathsf{i} \in \mathrm{N}}$, $\mathsf{p_i}(\boldsymbol{\theta}) = \mathrm{p}_\mathsf{i}$, and values p_i, $x_{\mathsf{i},j}$ are constants (0-ary functions) in $\mathcal{F}$.

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

$$\varphi_{\mathrm{impl}}(\mathsf{s}, \mathrm{E}, \boldsymbol{\theta}) =_{\mathrm{def}} \exists \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}(\mathrm{terminal} \wedge \varphi_{\mathsf{s}(\boldsymbol{\theta})})$$

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

Theorem 5.1. *A mechanism* $\mathcal{G}$ E*-implements a SCF* s *iff for every type profile* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $[\![\varphi_{\mathrm{impl}}(\mathsf{s}, \mathrm{E}, \boldsymbol{\theta})]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$.

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

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

First, because the $\mathsf{SLK}[\mathcal{F}]$ formula $\mathbf{E}(\boldsymbol{s})$ correctly characterises **E**-equilibria (lemma 5.1 and 5.2), letting $\mathcal{A}_{\boldsymbol{\sigma}} : s_\mathsf{i} \mapsto \sigma_\mathsf{i}$ we have that $[\![\mathrm{E}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v_\iota^{\boldsymbol{\theta}}) = 1$.

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

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

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

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 s, and $[\![\varphi_{\mathrm{impl}}(\mathsf{s}, \mathrm{E}, \boldsymbol{\theta})]\!]^{\mathcal{G}}_{\mathcal{A}}(v^{\boldsymbol{\theta}}_{\iota}) = -1$. □

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 s, 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

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

5.3.4 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 $\mathcal{B}_{\mathsf{i}} = \Theta_{\mathsf{i}}$ for each i. 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).

Strategyproofness 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 $\hat{\theta}_{\mathsf{i}}$ be the truth-revealing strategy for i, defined as $\hat{\theta}_{\mathsf{i}}(v^{\boldsymbol{\theta}}_{\iota}) = \theta_{\mathsf{i}}$.

Definition 5.7. A direct-revelation mechanism $\mathcal{G}$ is *strategy-proof* if $(\hat{\theta}_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}$ is a dominant strategy equilibrium from $v^{\boldsymbol{\theta}}_{\iota}$, for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

Strategyproofness 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 5.1. *A direct-revelation mechanism $\mathcal{G}$ is strategyproof iff* $[\![\mathrm{DSE}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$ *for all* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$*, where* $\mathcal{A}(s_{\mathsf{i}}) = \hat{\theta}_{\mathsf{i}}$ *for each* i.

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

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 nonnegative utility. Hereafter we express in $\mathsf{SLK}[\mathcal{F}]$ the notion of (ex-post) individual rationality (Nisan et al., 2007).

Definition 5.8. A SCF $\mathsf{s} = (\mathsf{x}, \{\mathsf{p}_{\mathsf{i}}\})$ is *individually rational* if for every $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $\mathsf{v}_{\mathsf{i}}(\mathsf{x}(\boldsymbol{\theta})) - \mathsf{p}_{\mathsf{i}}(\boldsymbol{\theta}) \geq 0$ for each agent i.

Let us define the following formula:

$$\mathrm{IR} =_{\mathrm{def}} \bigwedge_{\mathsf{i} \in \mathrm{N}} 0 \leq \mathrm{util}_{\mathsf{i}}$$

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

Proposition 5.2. *Let* s *be a SCF,* $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$*, and* $\mathcal{G}$ *a mechanism that* E*-implements* s*.* s *is individually rational iff* $[\![\text{Capture-alt}(\mathsf{s}, \mathrm{E}, \boldsymbol{\theta}, \mathrm{IR})]\!]^{\mathcal{G}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$ *for all* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$*.*

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

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

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 5.9. A social choice function $\mathsf{s} = (\mathsf{x}, \{\mathsf{p}_{\mathsf{i}}\})$ is *allocatively efficient* if for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$,

$$\sum_{\mathsf{i} \in \mathrm{N}} \mathsf{v}_{\mathsf{i}}(\mathsf{x}(\boldsymbol{\theta}), \theta_{\mathsf{i}}) = \max_{x \in \mathcal{X}} \sum_{\mathsf{i} \in \mathrm{N}} \mathsf{v}_{\mathsf{i}}(x, \theta_{\mathsf{i}})$$

Define formula

$$\text{Eff} =_{\text{def}} \sum_{\mathsf{i}\in N} \mathsf{v}_\mathsf{i}(\text{all}_{1,1}, \ldots, \text{all}_{\mathsf{n},\mathsf{m}}, \text{type}_\mathsf{i}) = \text{maxv}_{\boldsymbol{\theta}}$$

where, for each $\boldsymbol{\theta}$, $\text{maxv}_{\boldsymbol{\theta}} = \max_{x\in\mathcal{X}} \sum_{\mathsf{i}\in N} \mathsf{v}_\mathsf{i}(x, \theta_\mathsf{i})$ 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 s is efficient by verifying the satisfaction value of the formula Eff in a mechanism that implements s.

Proposition 5.3. *Let* s *be a SCF,* $\text{E} \in \{\text{NE}, \text{DSE}\}$*, and* $\mathcal{G}$ *a mechanism that* E*-implements* s*.* s *is allocativelly efficient iff* $[\![\text{Capture-alt}(\mathsf{s}, \text{E}, \boldsymbol{\theta}, \text{Eff})]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$ *for all* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$*.*

Proof. Analogous to the proof of Proposition 5.2. □

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 5.10. A social choice function $\mathsf{s} = (\mathsf{x}, \{\mathsf{p}_\mathsf{i}\})$ is *strongly budget-balanced* (resp., *weakly budget-balanced*) if $\sum_{\mathsf{i}\in N} \mathrm{p}_\mathsf{i}(\boldsymbol{\theta}) = 0$ (resp., $\sum_{\mathsf{i}\in N} \mathrm{p}_\mathsf{i}(\boldsymbol{\theta}) \geq 0$) for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

Define formula

$$\text{SBB} =_{\text{def}} 0 = \sum_{\mathsf{i}\in N} \text{pay}_\mathsf{i}$$

and define WBB similarly, with $\leq$ instead of $=$.

Again, we can prove that

Proposition 5.4. *Let* s *be a SCF,* $\text{E} \in \{\text{NE}, \text{DSE}\}$*, and* $\mathcal{G}$ *a mechanism that* E*-implements* s*. It holds that* s *is* SBB *iff* $[\![\text{Capture-alt}(\mathsf{s}, \text{E}, \boldsymbol{\theta}, \text{SBB})]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$ *for all* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$*, and similarly for* WBB*.*

Proof. Analogous to the proof for Proposition 5.2. □

Example 5.5. 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 5.3, with the first-price social choice function s_{fp}, for any type profile $\boldsymbol{\theta}$ we have $[\![\text{Capture-alt}(\mathsf{s}_{\text{fp}}, \text{NE}, \boldsymbol{\theta}, \text{IR})]\!]^{\mathcal{G}_{\text{dut}}}(v_\iota^{\boldsymbol{\theta}}) = 1$ and $[\![\text{Capture-alt}(\mathsf{s}_{\text{fp}}, \text{NE}, \boldsymbol{\theta}, \text{WBB})]\!]^{\mathcal{G}_{\text{dut}}}(v_\iota^{\boldsymbol{\theta}}) = 1$.

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 5.11. A social choice function $\mathsf{s} = (\mathsf{x}, \{\mathsf{p_i}\})$ is *Pareto optimal* if, for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, for all $\mathsf{i} \in \mathrm{N}$ and for all $\alpha \neq \mathsf{s}(\boldsymbol{\theta})$, if $u_\mathsf{i}(\alpha, \theta_\mathsf{i}) > u_\mathsf{i}(\mathsf{s}(\boldsymbol{\theta}), \theta_\mathsf{i})$ then there exists an agent $\mathsf{r} \in Ag$ such that $u_\mathsf{r}(\alpha, \theta_\mathsf{r}) < u_\mathsf{r}(\mathsf{s}(\boldsymbol{\theta}), \theta_\mathsf{r})$.

For every alternative $\alpha \in \mathrm{Alt}$, every type profile $\boldsymbol{\theta}$ and agent i, let $\mathrm{utilalt}_{\mathsf{i},\alpha} : \theta_\mathsf{i} \mapsto u_\mathsf{i}(\alpha, \theta_\mathsf{i})$ be a function in $\mathcal{F}$. Define formula (recall formula util_i, defined in Section 5.3.3):

$$\mathrm{PO} =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N},\alpha\in\mathrm{Alt}} (\mathrm{util}_\mathsf{i} < \mathrm{utilalt}_{\alpha,\mathsf{i}}(\mathrm{type}_\mathsf{i}) \rightarrow (\bigvee_{\mathsf{r}\in\mathrm{N}} \mathrm{utilalt}_{\alpha,\mathsf{r}}(\mathrm{type}_\mathsf{i}) < \mathrm{util}_\mathsf{r}))$$

As for Proposition 5.2, we can prove:

Proposition 5.5. *Let* s *be a SCF,* $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$*, and* $\mathcal{G}$ *a mechanism that* E*-implements* s*. It holds that* s *is PO iff* $[\![\text{Capture-alt}(\mathsf{s}, \mathrm{E}, \boldsymbol{\theta}, \mathrm{PO})]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$ *for all* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$*.*

Proof. Analogous to the proof of Proposition 5.2. □

5.3.5 Revenue Benchmarks with Knowledge

Let us now go further by considering the interplay between the agents' epistemic state and mechanism properties. To do so, we focus on the *auctioneer's revenue*, *i.e.*, the total payment among the agents. Guaranteeing a revenue is an important issue in Mechanism Design (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 $\mathcal{B}_\mathsf{i} \subset \boldsymbol{\Theta}$ denotes a belief for agent i 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:

$$2^{\mathrm{nd}}(\boldsymbol{\mathcal{B}}) =_{\mathrm{def}} \text{2nd-max}(\mathrm{smv}_{\mathsf{i}_1}(\boldsymbol{\mathcal{B}}), \ldots, \mathrm{smv}_{\mathsf{i}_\mathsf{n}}(\boldsymbol{\mathcal{B}}))$$

where $\mathrm{smv}_\mathsf{i}(\boldsymbol{\mathcal{B}}) =_{\mathrm{def}} \min_{\boldsymbol{\theta}\in B_\mathsf{i}}(\max_{\mathsf{r}\in\mathrm{N}}(\theta_\mathsf{r}))$ denotes the *sure maximum value* according to i.

Let $\mathcal{G}$ be a mechanism. To each position $v \in V$ we can associate a correct belief $\mathcal{B}_{\mathsf{i}}(v)$ for each agent i as follows: $\mathcal{B}_{\mathsf{i}}(v) =_{\text{def}} (\{\ell(v', \text{type}_b) : v' \sim_{\mathsf{i}} v\})_{\mathsf{r}\in\mathrm{N}}$. We then let $\boldsymbol{\mathcal{B}}(v) = (\mathcal{B}_{\mathsf{i}}(v))_{\mathsf{i}\in\mathrm{N}}$. 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:

$$\varphi^{\text{smv}}_{\mathsf{i}} =_{\text{def}} K_{\mathsf{i}} \max_{\mathsf{i}'\in\mathrm{N}}(\text{type}_{\mathsf{i}'})$$
$$\varphi_{2\text{nd}} =_{\text{def}} 2\text{nd-max}(\varphi^{\text{smv}}_{\mathsf{i}_1}, \ldots, \varphi^{\text{smv}}_{\mathsf{i}_n})$$

It follows directly that:

Proposition 5.6. *Given a mechanism $\mathcal{G}$, a position v and a belief profile $\boldsymbol{\mathcal{B}}(v)$, it holds that $[\![\varphi_{2\text{nd}}]\!]^{\mathcal{G}}(v) = 2^{nd}(\boldsymbol{\mathcal{B}}(v))$.*

Proof. Fix an assignment $\mathcal{A}$. We have that $[\![\varphi_{2\text{nd}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = [\![2\text{nd-max}(\varphi^{\text{smv}}_{\mathsf{i}_1}, \ldots, \varphi^{\text{smv}}_{\mathsf{i}_n})]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$. By the semantics of K_{i}, $\varphi^{\text{smv}}_{\mathsf{i}}$ denotes the minimum value of $[\![\max_{\mathsf{i}\in\mathrm{N}}(\text{type}_{\mathsf{i}})]\!]^{\mathcal{G}}_{\mathcal{A}}(v')$, for all $v' \sim_{\mathsf{i}} v$. Since $\mathcal{B}_{\mathsf{i}}(v) = (\{\ell(v', \text{type}_b) : v' \sim_{\mathsf{i}} v\})_{\mathsf{r}\in\mathrm{N}}$, it holds that $[\![\varphi^{\text{smv}}_{\mathsf{i}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = \min_{\boldsymbol{\theta}\in B_{\mathsf{i}}}(\max_{\mathsf{r}\in\mathrm{N}}(\theta_{\mathsf{r}}))$. Therefore, $[\![\varphi_{2\text{nd}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 2\text{nd-max}(\text{smv}_{\mathsf{i}_1}(\boldsymbol{\mathcal{B}}), \ldots, \text{smv}_{\mathsf{i}_n}(\boldsymbol{\mathcal{B}}))$ or simply $[\![\varphi_{2\text{nd}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 2^{\text{nd}}(\boldsymbol{\mathcal{B}}(v))$. □

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 $\varepsilon > 0$ 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

$$\varphi_{\text{revenue}} =_{\text{def}} \mathbf{F}(\text{terminal} \wedge \sum_{\mathsf{i}\in\mathrm{N}}(\mathsf{p}_{\mathsf{i}}))$$
$$\varphi_{\text{last}-2\text{nd}} =_{\text{def}} \mathbf{F}(\mathbf{X}\text{terminal} \wedge \varphi_{2\text{nd}})$$

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 $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$, one can check whether the following formula has value 1 on this mechanism:

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

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 5.3.2.

5.4 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 5.12. The *model-checking problem* for $\mathsf{SLK}[\mathcal{F}]$ consists in deciding, given a sentence φ, wCGSii $\mathcal{G}$, position v in $\mathcal{G}$ and predicate $P \subseteq [-1, 1]$, whether $[\![\varphi]\!]^{\mathcal{G}}(v) \in P$.

For quantitative LTL ($\mathsf{LTL}[\mathcal{F}]$) model checking is in Pspace (Almagor et al., 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 5.2. *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 $O((|\text{free}(\varphi)| + |\text{N}|) \cdot |V| \cdot \log |\mathcal{B}|)$. Next, for the base case it is clear that $[\![p]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$ can be computed in constant space. For strategy quantification $[\![\exists s_{\mathsf{i}}.\, \varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$, besides the recursive call to $[\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[s\mapsto\sigma]}(v)$ we need space $O(|V| \cdot \log |\mathcal{B}|)$ to store the current strategy and the current maximum value computed. The case for $[\![K_{\mathsf{i}}\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$ is clear. For $[\![f(\varphi_1, \ldots, \varphi_m)]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$, 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 $\rho_1 = v_0 \ldots v_{k-1}$ and $\rho_2 = v_k \ldots v_{l-1}$ such that $\pi = \rho_1 \cdot \rho_2^{\omega}$. It follows that

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

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 key 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.

5.5 Conclusion

In this chapter, 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 due to our focus on auctions, but it is enough to replace them with abstract choices to capture any kind of deterministic mechanisms. In addition we showed how $\mathsf{SLK}[\mathcal{F}]$ allows capturing properties that are central in the design of many types of mechanisms other than auctions, including efficiency and Pareto optimality.

We considered the setting of imperfect information about the agents' types. One line for future work is to explore mechanisms with *incomplete information* (that is, Bayesian mechanisms), in which there is a commonly known probability distribution over the possible types for each agent.

The present setting is enough to model many kinds of auctions where memoryless strategies are sufficient to represent the bidders' behavior, such as one-shot or English auctions. However, when participating in sequential auctions, agents could gather information by observing other agents' and act based on what happened in previous steps of the game (Jeitschko, 1998). In the next chapter we explore such scenario and investigate strategic reasoning in mechanism design when the participants have limited resources and, in special, bounded memory.

Chapter 6

Mechanisms and Natural Strategies

As discussed on the previous chapter, a number of logic-based languages have been recently introduced to reason about the strategic abilities of autonomous agents in multi-agent systems (MAS). Still, verification tools and techniques are comparatively less developed for *data-driven* and *data-intensive* systems[1], that is, contexts where the data content of processes, or *agents*, is key to model and account for the evolution of the system (Belardinelli et al., 2014; Montali et al., 2014). This is the case also for online advertising, where search engines sell ad placements for keywords continuously through auctions. This problem can be seen as an infinitely repeated game since the auction is executed whenever a user performs a query with the keyword. As advertisers may frequently change their bids, the game will have a large set of equilibria with potentially complex strategies, thus making the specification and verification of keyword auctions a complex problem to solve for current model checking methods. This issue is stressed by Edelman et al. (2007):

> "In principle, the sets of equilibria in such repeated games can be very large, with players potentially punishing each other for deviations. The strategies required to support such equilibria are usually quite complex, however, requiring precise knowledge of the environment and careful implementation. In theory, advertisers could implement such strategies via automated robots, but in practice they may not be able to: bidding software must first be authorized by the search engines, and search engines are unlikely to permit strategies that would allow advertisers to collude and substantially reduce revenues."

In this chapter, we investigate the use of *natural strategies* for reasoning about equilibria in keyword auctions. We directly build on the research by (Jamroga et al., 2019a,b) on natural strategies. We generalize their approach by considering quantitative semantics for both natural strategies and the logic, which is more suitable for reasoning about mechanisms with monetary transfer (*e.g.*, auctions). We also consider SL instead of ATL, due to its expressive power.

In our case, the bidding strategy in an auction should be executable for a *simple* artificial agent, as well as reasonably transparent to the human user, which makes natural strategies a good match. Moreover, natural strategies provide a way to

[1] "[Model checking] is mainly appropriate to control-intensive applications and less suited for data-intensive applications" (Baier and Katoen, 2008, p. 15).

define complexity (and hence also "simplicity") metrics for various functionality, security, and usability properties in MAS. By focusing on simple strategies, one can make the verification of equilibrium properties decidable, or even tractable, despite the prohibitive complexity of the general problem. This is especially evident for strategies with memory, which normally make the synthesis and model checking problems undecidable (Dima and Tiplea, 2011; Vester, 2013).

By leveraging on natural strategies, we introduce SL with quantitative semantics and natural strategies and imperfect information, denoted $\mathsf{NatSL}[\mathcal{F}]$. In a first step, we show how to model strategies for repeated keyword auctions and take advantage of the model for proving properties evaluating this game. In a second step, we study the logic in relation to the distinguishing power, expressivity, and model-checking complexity for strategies with and without recall.

6.1 Natural Strategies

We first present the notion of uniform natural strategies from (Jamroga et al., 2019b). Natural strategies are conditional plans, represented through an ordered list of condition-action rules (Jamroga et al., 2019b). The intuition is that the first rule whose condition holds in the history of the game is selected, and the corresponding action is executed. As we are considering the setting of imperfect information, the conditions are regular expressions over *weighted epistemic (WE) formulas*.

Given an agent i, the WE formulas over Φ, denoted $WE(\Phi)$, are prefixed by K_i and then possibly combined by functions in $\mathcal{F}$. That is, formulas in $WE(\Phi)$ are conditions on i's knowledge and are expressed by the following BNF:

$$\psi ::= \top \mid K_\mathsf{i}\varphi \mid f(\psi, \ldots, \psi)$$

$$\varphi ::= p \mid f(\varphi, \ldots, \varphi) \mid K_\mathsf{r}\varphi$$

where $f \in \mathcal{F}$ is a function, $p \in \Phi$ is an atomic proposition and $\mathsf{r} \in \mathrm{N}$ is an agent.

The semantics of natural strategies (and, as we shall see, $\mathsf{NatSL}[\mathcal{F}]$) are interpreted over concurrent game structures (Jamroga et al., 2019b). Such structures are similar to the ones defined in Chapter 5, but include a legality function describing the availability of actions in each state.

Definition 6.1. A *weighted concurrent game structure with imperfect information and legality* (wCGSl) is a tuple $\mathcal{G} = (\mathcal{B}, V, \mathrm{L}, \delta, \ell, V_\iota, \{\sim_\mathsf{i}\}_{\mathsf{i}\in\mathrm{N}})$ where:

- $\mathcal{B}$ is a finite set of *actions*;
- $\mathrm{L} : \mathrm{N} \times V \to 2^{\mathcal{B}}$ is a *legality function*, defining the availability of actions;
- δ is a transition function assigning a successor state $v' = \delta(v, (a_\mathsf{i})_{\mathsf{i}\in\mathrm{N}})$ to each state $v \in V$ and any tuple of actions $(a_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$, where $a_\mathsf{i} \in \mathrm{L}(\mathsf{i}, v)$;

and the remaining components are defined exactly as the homonyms components in wCGSii (see Definition 5.2).

We require that the wCGSl is uniform, that is $v \sim_{\mathsf{i}} v'$ implies $\mathrm{L}(\mathsf{i}, v) = \mathrm{L}(\mathsf{i}, v')$. In a state $v \in V$, each player i chooses an available action $a_{\mathsf{i}} \in \mathrm{L}(\mathsf{i}, v)$, and the game proceeds to state $\delta(v, \boldsymbol{a})$ where $\boldsymbol{a}$ is the *action profile* $(a_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}$.

Plays A *play* $\pi = v_0 v_1 v_2 \ldots$ is an infinite sequence of states such that for every $i \geq 0$ 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 state at index i in play π.

History A *history* $h = v_0 v_1 v_2 \ldots v_n$ is a finite sequence of states. The last element of a history is denoted by $last(h) = v_n$. Finally, $H_{\mathcal{G}}$ denotes the set of all histories in the wCGSl $\mathcal{G}$.

Given a wCGSl $\mathcal{G}$, a state $v \in V$ and a $WE(\Phi)$ formula φ, we inductively define the satisfaction value of φ in v, denoted $[\![\varphi]\!](v)$, as follows:

$$
\begin{aligned}
[\![p]\!](v) &= \ell(v, p) \\
[\![K_{\mathsf{i}} \varphi]\!](v) &= \min_{v' \sim_{\mathsf{i}} v} [\![\varphi]\!](v') \\
[\![f(\varphi_1, \ldots, \varphi_m)]\!](v) &= f([\![\varphi_1]\!](v), \ldots, [\![\varphi_m]\!](v))
\end{aligned}
$$

The semantics for the knowledge modality is the standard in the literature on fuzzy epistemic logic (see, for instance, (Maruyama, 2021) and Chapter 5). Let $Reg(WE(\Phi))$ be the set of regular expressions over the weighted epistemic conditions $WE(\Phi)$, defined with the constructors $\cdot, \cup, ^*$ representing concatenation, nondeterministic choice, and finite iteration, respectively. Given a regular expression r and the language $\mathcal{L}(r)$ on words generated by r, a history h is *consistent* with r iff there exists $b \in \mathcal{L}(r)$ such that $|h| = |b|$ and $[\![b[i]]\!](h[i]) = 1$, for all $0 \leq i \leq |h|$. Intuitively, a history h is consistent with a regular expression r if the i-th weighted epistemic condition in r "holds" in the i-th state of h (for any position i in h).

A *uniform natural strategy with recall* σ_{i} for agent i is a sequence of pairs (r, a), where $r \in Reg(WE(\Phi))$ is a regular expression, and a is an action available in $\mathrm{last}(h)$, for all histories $h \in H_{\mathcal{G}}$ consistent with r. The last pair on the sequence is required to be $(\top^*, a)$, with $a \in \mathrm{L}(\mathsf{i}, v)$ for every $v \in V$ and some $a \in \mathcal{B}$.

A *uniform memoryless natural strategy* is a special case of natural strategy in which each condition is a weighted epistemic formula (i.e., no regular operators are allowed).

Natural strategies are uniform in the sense they specify the same actions in indistinguishable states (see (Jamroga et al., 2019b)). We define Str^{ρ}_{i} to be the set of uniform natural strategies for agent i and $Str^{\rho} = \cup_{\mathsf{i} \in \mathrm{N}} Str^{\rho}_{\mathsf{i}}$, where $\rho \in \{ir, iR\}$[2].

Let $size(\sigma_{\mathsf{i}})$ denote the number of guarded actions in σ_{i}, $cond_i(\sigma_{\mathsf{i}})$ be the i-th guarded condition on σ_{i}, $cond_i(\sigma_{\mathsf{i}})[j]$ be the j-th WE formula of the guarded

[2]As usual in the verification process, we denote imperfect recall with r, perfect recall with R, imperfect information with i, and perfect information with I.

condition σ_{i}, and $act_i(\sigma_{\mathsf{i}})$ be the corresponding action. Finally, $match(h, \sigma_{\mathsf{i}})$ is the smallest index $i \leq size(\sigma_{\mathsf{i}})$ such that for all $0 \leq j \leq |last(h)|$, $[\![cond_i(\sigma_{\mathsf{i}})[j]]\!](h[j]) = 1$[3] and $act_i(\sigma_{\mathsf{i}}) \in \mathrm{L}(\mathsf{i}, last(h))$. In other words, $match(h, \sigma_{\mathsf{i}})$ matches the state $last(h)$ with the first condition in σ_{i} that holds in h, and action available in $last(h)$.

Measurement of Natural Strategies. The complexity of the strategy σ is the total size of its representation and is denoted as follows:

$$compl(\sigma) =_{\mathrm{def}} \sum_{(r,a)\in\sigma} |r|$$

where $|r|$ is the number of symbols in r, except by parentheses. If r is a n-ary function in $\mathcal{F}$, then $|r| = n + 1$. See (Jamroga et al., 2019a) for other metrics for the measuring the complexity of natural strategies.

6.2 Quantitative Natural Strategy Logic

Standard $\mathsf{SL}[\mathcal{F}]$ (Bouyer et al., 2019) considers combinatorial strategies, that is, strategies are functions mapping histories (or states) to actions. For reasoning about intuitive and simple strategies, we introduce $\mathsf{SL}[\mathcal{F}]$ with natural strategies and imperfect information, denoted $\mathsf{NatSL}[\mathcal{F}]$. $\mathsf{SL}[\mathcal{F}]$ and $\mathsf{NatSL}[\mathcal{F}]$ have similar syntax and semantics, the main difference is that the quantification of strategies in $\mathsf{NatSL}[\mathcal{F}]$ is defined for natural strategies with bounded complexity.

Throughout this chapter, let $\rho \in \{ir, iR\}$ denote whether the semantics considers memoryless or recall strategies.

Assignments. An *assignment* $\mathcal{A} : \mathrm{N} \cup \mathrm{Var} \to Str^{\rho}$ is a function from players and variables to strategies. For an assignment $\mathcal{A}$, an agent i and a strategy σ for i, $\mathcal{A}[\mathsf{i} \mapsto \sigma]$ is the assignment that maps i 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 state 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 $v_0 v_1 \ldots$ such that $v_0 = v$ and for all $i \geq 0$, $v_{i+1} = \delta(v_i, \boldsymbol{a})$ where for all $\mathsf{i} \in \mathrm{N}$, $\boldsymbol{a}_{\mathsf{i}} = act_{match(v_i, \mathcal{A}(\mathsf{i}))}(\mathcal{A}(\mathsf{i}))$.

Definition 6.2. The syntax of $\mathsf{NatSL}[\mathcal{F}]$ is defined as follows:

$$\varphi ::= p \mid \exists s_{\mathsf{i}}^{\leq k}. \varphi \mid (\mathsf{i}, s_{\mathsf{i}})\varphi \mid f(\varphi, \ldots, \varphi) \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi$$

[3]Note that, we considered the case in which the condition have the same length of the history. There is also the case in which the condition is shorter than the history. This is due to the usage of the finite iteration operator. In the latter case, we need to check a finite number of times the same weighted epistemic formula in different states of the history. For more details on this aspect see (Jamroga et al., 2019a,b).

where $p \in \mathrm{AP}$, $s_{\mathsf{i}} \in \mathrm{Var} \cup Str^{\rho}_{\mathsf{i}}$, $k \in \mathbb{Z}$, $\mathsf{i} \in \mathrm{N}$, and $f \in \mathcal{F}$.

The intuitive reading of the operators is as follows: $\exists s_{\mathsf{i}}^{\leq k}.\,\varphi$ means that there exists a strategy with complexity less or equal than k for agent i such that φ holds. The remaining operators correspond to the ones defined for $\mathsf{SLK}[\mathcal{F}]$ (see Chapter 5).

A variable is *free* in formula φ if it is bound to an agent without being quantified upon, and an agent i 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 i. The set of free variables and agents in φ is written $\mathrm{free}(\varphi)$, and a formula φ is a *sentence* if $\mathrm{free}(\varphi) = \emptyset$. The strategy quantifier $\exists s_{\mathsf{i}}^{\leq k}.\,\varphi$ quantifies on strategies *for agent* i.

Definition 6.3. Let $\mathcal{G} = (\mathcal{B}, V, \delta, \ell, V_\iota, \{\sim_{\mathsf{i}}\}_{\mathsf{i}\in\mathrm{N}})$ be a wCGSl, and $\mathcal{A}$ an assignment. The satisfaction value $[\![\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) \in [-1, 1]$ of a $\mathsf{NatSL}[\mathcal{F}]$ formula φ in a state v is defined as follows, where π denotes $\mathrm{Out}(v, \mathcal{A})$:

$$
\begin{aligned}
[\![p]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) &= \ell(v, p)\\
[\![\exists s_{\mathsf{i}}^{\leq k}.\,\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) &= \max_{\sigma\in\{\alpha\in Str^{\rho}_{\mathsf{i}}:compl(\alpha)\leq k\}} [\![\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}[s_{\mathsf{i}}\mapsto\sigma]}(v)\\
[\![(\mathsf{i}, s_{\mathsf{i}})\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) &= [\![\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}[\mathsf{i}\mapsto\mathcal{A}(s_{\mathsf{i}})]}(v) \text{ if } s_{\mathsf{i}} \in \mathrm{Var}\\
[\![(\mathsf{i}, \sigma_{\mathsf{i}})\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) &= [\![\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}[\mathsf{i}\mapsto\sigma_{\mathsf{i}}]}(v) \text{ if } \sigma_{\mathsf{i}} \notin \mathrm{Var}\\
[\![f(\varphi_1,\ldots,\varphi_m)]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) &= f([\![\varphi_1]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v),\ldots,[\![\varphi_m]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v))\\
[\![\mathbf{X}\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) &= [\![\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(\pi_1)\\
[\![\varphi_1\mathbf{U}\varphi_2]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) &= \sup_{i\geq 0} \min\Big([\![\varphi_2]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(\pi_i), \min_{0\leq j<i} [\![\varphi_1]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(\pi_j)\Big)
\end{aligned}
$$

If φ is a sentence, its satisfaction value does not depend on the assignment, and we write $[\![\varphi]\!]^{\mathcal{G},\rho}(v)$ for $[\![\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v)$ where $\mathcal{A}$ is any assignment. We also let $[\![\varphi]\!]^{\mathcal{G},\rho} = \min_{v_\iota\in V_\iota}[\![\varphi]\!]^{\mathcal{G},\rho}(v_\iota)$. The abbreviations $\bot$, $\varphi \to \varphi'$, $\varphi \wedge \varphi'$, $\mathbf{F}\psi$, $\mathbf{G}\psi$, and $\forall s_{\varphi}^{\leq k}$ are defined in the standard way (see Section 7.1).

In the next section, we show how to use $\mathsf{NatSL}[\mathcal{F}]$ for reasoning about mechanisms by considering the case of repeated keyword auctions.

6.3 Repeated Keyword Auctions

Modeling mechanisms with monetary transfer and private valuations require handling quantitative features and imperfect information. Memoryless strategies are enough for mechanisms in which all relevant information is encoded in the current state (*e.g.*, English auction). In repeated auctions, agents may, as well, use information from the previous states for choosing their strategies.

We now focus on using $\mathsf{NatSL}[\mathcal{F}]$ to model and verify repeated keyword auctions and related strategies. Repeated keyword auctions are used by online search engines, such as Google and Yahoo, for selling advertising slots when users perform a search with a keyword Cary et al. (2007). For example, if a store buys a slot for the keyword

"computer", when a user searches this term, the store's sponsored link will be shown for her. For a keyword of interest, the advertisers (bidders) submit a bid stating the maximum amount she is willing to pay for a click on her sponsored link. When a user submits a query, an auction is run to determinate the slot allocation among the advertisers bidding on the keyword of interest. The most common mechanism for keyword auctions is the Generalized Second Price (GSP) Cary et al. (2007), in which the agents are allocated slots in decreasing order of bids and the payment for the slot s is the bid of the agent allocated to the slot $s+1$.

We assume that $\mathcal{F}$ contains the function $\leq : (x, y) \mapsto 1$ if $x \leq y$ and $\leq : (x, y) \mapsto -1$ otherwise; and for readability we use the infix notation $x \leq y$ in the formula. We also assume that $\mathcal{F}$ contains the equality $=$ and comparison functions $<$, $>$, $\geq$ (defined similarly). Finally, we assume $\mathcal{F}$ contains functions $-$, $\sum$, $\times$, $\setminus$, min, max and $argmax$ with the standard meaning (for details, see Chapter 5).

Let us fix a price increment $\mathsf{inc} \in (0, 1]$, a set of slots $S = \{1, \ldots, \mathsf{m}\}$, where $\mathsf{m} \in \mathbb{N} \setminus \{0\}$. Each slot has a click-through rate $\theta_1 > \ldots > \theta_{\mathsf{m}}$, where $\theta_s \in [0, 1]$ is the probability that the user will click on the advertisement in slot s. The agents in N are the advertisers, each one having a private valuation $\mathsf{v_i} \in \mathsf{V_i}$ for a click, where $\mathsf{V_i} \subset [0, 1]$ is a finite set of possible valuations. We assume the valuations are distinct, that is, if $\mathsf{i} \neq \mathsf{i}'$, then $\mathsf{v_i} \neq \mathsf{v_{i'}}$. We denote by $\prec$ an arbitrary order among the agents in N, used in case of ties. The atomic propositional set is $\Phi = \{\mathrm{all}_{\mathsf{i},s}, \mathrm{pay}_s, \vartheta_{\mathsf{i}} : \mathsf{i} \in \mathrm{N}, s \in S\}$, where $\mathrm{all}_{\mathsf{i},s}$ represents whether agent i is allocated to slot s, pay_s denotes the price of slot s and ϑ_{i} denotes i's valuation. Define $\mathcal{G}_{GSP} = (\mathcal{B}, V, \mathrm{L}, \delta, \ell, V_\iota, \{\sim_{\mathsf{i}}\}_{\mathsf{i} \in \mathrm{N}})$, where:

- $\mathcal{B} = \{0 + x \times \mathsf{inc} : 0 \leq x \leq \frac{1}{\mathsf{inc}}\}$, where $b \in \mathcal{B}$ denotes a bid with price b for a click; given $\boldsymbol{a} = (a_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}$, let $rank_{\boldsymbol{a}} = (\mathsf{i}_1, \ldots, \mathsf{i}_{\mathsf{n}})$ be the sequence of distinct agents in N ordered by their bid, that is, $i < j$ if $a_{\mathsf{i}_i} > a_{\mathsf{i}_j}$ or $a_{\mathsf{i}_i} = a_{\mathsf{i}_j}$ and $\mathsf{i}_i \prec \mathsf{i}_j$ for $i, j \in \{1, \ldots, \mathsf{n}\}$ with $i \neq j$. In case of draws, the sequence is determined with respect to $\prec$. We let $rank_{\boldsymbol{a}}(i)$ denote the agent in the i-th position of the sequence $rank_{\boldsymbol{a}}$.

- $V = \{\langle al_1, \ldots, al_{\mathsf{m}}, pr_1, \ldots, pr_{\mathsf{m}}, (vl_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}\rangle : al_s \in \mathrm{N} \cup \{none\}$ & $pr_s \in \mathcal{B}$ & $vl_{\mathsf{i}} \in \mathsf{V_i}$ & $\mathsf{i} \in \mathrm{N}$ & $1 \leq s \leq \mathsf{m}\}$, where each state represents the current slot allocation and prices, with al_s, pr_s, and vl_{i} denoting the winner of slot s, the price per click of s and i's valuation, resp.;

- For each $\mathsf{i} \in \mathrm{N}$ and $v \in V$, $\mathrm{L}(\mathsf{i}, v) = \mathcal{B}$;

- For each $v \in V$ and $\boldsymbol{a} = (a_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}$ such that $a_{\mathsf{i}} \in \mathrm{L}(\mathsf{i}, v)$, the transition function uses the agent's bids to chose the next allocations and prices and is defined as follows: $\delta(v, (a_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}) = \langle al'_1, \ldots, al'_{\mathsf{m}}, pr'_1, \ldots, pr'_{\mathsf{m}}, (vl_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}\rangle$, where for each agent i and slot s,

$$al_s = \begin{cases} rank_{\boldsymbol{a}}(s) & \text{if } s \leq \mathsf{n} \\ none & \text{otherwise} \end{cases}$$

$$pr_s = \begin{cases} a_{rank_{\boldsymbol{a}}(s+1)} & \text{if } s+1 \leq \mathsf{n} \\ 0 & \text{otherwise} \end{cases}$$

- For each agent i, slot $s \in S$ and state $v = \langle al_1, \ldots, al_{\mathsf{m}}, pr_1, \ldots, pr_{\mathsf{m}}, (vl_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}\rangle$, the weight function is defined as follows:

  i. $\ell(v, \mathrm{all}_{\mathsf{i},s}) = 1$ if $al_s = \mathsf{i}$, and $\ell(v, \mathrm{all}_{\mathsf{i},s}) = 0$ otherwise;

  ii. $\ell(v, \mathrm{pay}_s) = pr_s$; and

  iii. $\ell(v, \vartheta_{\mathsf{i}}) = vl_{\mathsf{i}}$.

- In an initial state, the prices are 0 and the slots are allocated to *none*, that is, $V_\iota = \{\langle none, \ldots, none, 0, \ldots, 0, vl_1, \ldots, vl_{\mathsf{n}}\rangle \in V\}$;

- For each agent i and two states $v = \langle al_1, \ldots, al_{\mathsf{m}}, pr_1, \ldots, pr_{\mathsf{m}}, (vl_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}\rangle$ and $v' = \langle al'_1, \ldots, al'_{\mathsf{m}}, pr'_1, \ldots, pr'_{\mathsf{m}}, (vl'_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}\rangle$ in V, the observation relation $\sim_{\mathsf{i}}$ is such that if $v \sim_{\mathsf{i}} v'$ then (i) $al_s = al_{s'}$, for each $1 \leq s \leq \mathsf{m}$; (ii) $p_s = p_{s'}$, for each $1 \leq s \leq \mathsf{m}$; (iii) $vl_{\mathsf{i}} = vl'_{\mathsf{i}}$.

Notice there is exactly one initial state for each possible valuation profile in $(\prod_{\mathsf{i}\in\mathrm{N}} \mathsf{v}_{\mathsf{i}})$. Additionally, valuations remain unchanged after the initial state. We use the constant $\frac{1}{s}$ as the value in $[-1, 1]$ representing the slot s. The utility of agent i when she is assigned to slot s is denoted by the formula $\mathrm{util}_{\mathsf{i},s} =_{\mathrm{def}} \theta_s \times (\vartheta_{\mathsf{i}} - \mathrm{pay}_s)$. The utility for agent i in a state depends on her actual allocation, that is,

$$\mathrm{util}_{\mathsf{i}} =_{\mathrm{def}} \sum_{s\in S} \mathrm{all}_{\mathsf{i},s} \times \mathrm{util}_{\mathsf{i},s}$$

6.3.1 Solution Concepts for GSP

In this section, we show how $\mathsf{NatSL}[\mathcal{F}]$ can be used for the verification of mechanisms with natural strategies. Bearing in mind our motivating example, we aim at rephrasing conditions and properties usually considered in the analysis of keyword auctions (Cary et al., 2007; Edelman et al., 2007; Varian, 2007).

6.3.1.1 Nash equilibrium

Since auctions are noncooperative, the solution concept in the pure strategy setting usually considered is the Nash equilibrium (NE). As we previously discussed, NE captures the notion of stable solution: a strategy profile is NE if no player can improve her utility through an unilateral change of strategy (Nisan et al., 2007). With $\mathsf{NatSL}[\mathcal{F}]$, we restrict the range of strategies to simple ones, as it enables us to reason about artificial agents with limited capabilities and human-friendly strategies. Let $\boldsymbol{\sigma} = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ be a profile of strategies and $k > 0$ and define the formula

$$\mathrm{NE}(\boldsymbol{\sigma}, k) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N}} \forall t.^{\leq k} \left[(\mathrm{N}_{-\mathsf{i}}, \sigma_{-\mathsf{i}})(\mathsf{i}, t)\mathbf{X}\mathrm{util}_{\mathsf{i}} \leq (\mathrm{N}, \boldsymbol{\sigma})\mathbf{X}\mathrm{util}_{\mathsf{i}}\right]$$

The formula $\mathrm{NE}(\boldsymbol{\sigma}, k)$ means that, for every agent and alternative strategy t of complexity at most k, binding to t when everyone else binds to their strategies in $\boldsymbol{\sigma}$ leads to at most the same utility as when she also binds to her strategy in $\boldsymbol{\sigma}$. In relation to strategies with complexity at most k, the strategy profile $\boldsymbol{\sigma}$ leads to a NE in the next state of v if $[\![\mathrm{NE}(\boldsymbol{\sigma}, k)]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) = 1$.

Predicting outcomes of a keyword auction is a difficult task given the infinite nature of NE continuum (Yuan et al., 2017). For this reason, refined solution concepts have been proposed to reduce the NE continuum to subsets. Edelman *et al.* 2007 studied the subset called locally envy-free equilibrium (LEFE), in which no advertiser can improve her utility by exchanging her current slot to the one ranked one position above, given the current prices.

6.3.1.2 Locally envy free equilibrium

Let $\boldsymbol{\sigma} = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ be a profile of strategies, we define the formula

$$\mathrm{LEFE}(\boldsymbol{\sigma}) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N}} (\mathrm{N}, \boldsymbol{\sigma})\mathbf{X}[\mathrm{LEF}^{\mathsf{i}}_{wins} \wedge \mathrm{LEF}^{\mathsf{i}}_{loses}]$$

where $\mathrm{LEF}^{\mathsf{i}}_{wins} =_{\mathrm{def}} \bigwedge_{1<s\leq\mathsf{m}}(\mathrm{all}_{\mathsf{i},s} = 1 \rightarrow \mathrm{util}_{\mathsf{i},s} \geq \mathrm{util}_{\mathsf{i},s-1})$ indicates that when an agent is allocated to a slot, she does not prefer to switch to the slot right above and $\mathrm{LEF}^{\mathsf{i}}_{loses} =_{\mathrm{def}} (\bigwedge_{s\in S} \mathrm{all}_{\mathsf{i},s} = 0) \rightarrow 0 \geq \mathrm{util}_{\mathsf{i},\mathsf{m}}$ denotes that agents who were not assigned to any slot do not prefer to get the last slot.

$\mathrm{LEFE}(\boldsymbol{\sigma})$ means that, for any agent, when everyone follows the strategies in $\boldsymbol{\sigma}$, it holds that (i) if she wins s, her utility for s is greater than for slot $s-1$ (at current prices) and (ii) if she does not get any slot, then her utility for the last slot is at most zero. Strategy profile $\boldsymbol{\sigma}$ leads to LEFE in the next state of v if $[\![\mathrm{LEFE}(\boldsymbol{\sigma})]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v) = 1$.

Based on (Edelman et al., 2007; Varian, 2007), we have that any LEFE is also a NE:

Proposition 6.1. *For any complexity $k \geq 0$, state $v \in V$, $\rho \in \{iR, ir\}$ and strategy profile $\boldsymbol{\sigma} = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ with $\sigma_{\mathsf{i}} \in \mathrm{Str}^{\rho}_{\mathsf{i}}$ for each agent $\mathsf{i} \in \mathrm{N}$, $[\![\mathrm{LEFE}(\boldsymbol{\sigma}) \rightarrow \mathrm{NE}(\boldsymbol{\sigma}, k)]\!]^{\mathcal{G}_{GSP},\rho}(v) = 1$.*

Proof sketch. Let $\mathcal{A}$ be an assignment, $k \geq 0$ be a complexity bound for strategies, $v \in V$ be a state, $\rho \in \{iR, ir\}$, and $\boldsymbol{\sigma} = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ be a profile of ρ-strategies. Assume $[\![\mathrm{LEFE}(\boldsymbol{\sigma})]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v) = 1$. Let $v_{\boldsymbol{\sigma}} = \delta(v, \boldsymbol{b})$ where $\boldsymbol{b} = (b_{\mathsf{r}})_{\mathsf{r}\in\mathrm{N}}$ and $b_{\mathsf{r}} = act_{match(v,\sigma_{\mathsf{r}})}$ denotes the action performed by r in v if she follows σ_{r}. By definition, $\ell(v_{\boldsymbol{\sigma}}, \mathrm{all}_{\mathsf{r},s}) = 1$ iff $\mathsf{r} = rank_{\boldsymbol{b}}(s)$ is the winner of slot s and her payment is $\ell(v_{\boldsymbol{\sigma}}, \mathrm{pay}_s) = b_{rank_{\boldsymbol{b}}}(s{+}1)$. For each slot $s \in \{1, \ldots, min(\mathsf{m}, \mathsf{n})\}$, we consider whether its winner $\mathsf{i} = rank_{\boldsymbol{b}}(s)$ could improve her utility by deviating to strategy $t \in \{\alpha \in Str^{\rho}_{\mathsf{i}} : compl_{\rho}(\alpha) \leq k\}$. The case for agents who were not assigned any slot is proved similarly. Denote by $\bar{b} = match(v, t) \in \mathcal{B}$ the action that i would take if she followed t and $v_{(\sigma_{-\mathsf{i}},t)} = \delta(v, (b_{-\mathsf{i}}, \bar{b}))$ the next reached state from v when she follows t and others play according to σ.

If $s = 1$, any $\bar{b} \geq \ell(v_{\boldsymbol{\sigma}}, \text{pay}_s)$ does not change the outcome of the auction and $[\![(\text{N}_{-\mathsf{i}}, \sigma^{\rho}_{-\mathsf{i}})(\mathsf{i}, t)\mathbf{X}\text{util}_{\mathsf{i}} = (\text{N}, \boldsymbol{\sigma})\mathbf{X}\text{util}_{\mathsf{i}}]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v) = 1$. The same holds when $1 < s < m$ for $\bar{b} \in [\ell(v_{\boldsymbol{\sigma}}, \text{pay}_s), \ell(v_{\boldsymbol{\sigma}}, \text{pay}_{s-1})]$ and when $s = \mathsf{m}$ for $\bar{b} \leq \ell(v_{\boldsymbol{\sigma}}, \text{pay}_{\mathsf{m}})$. In the remaining cases, i would change her position in $rank_{\boldsymbol{b}}$ with other agent. By the results of Edelman et al. (2007) (see Lemma 1), the outcome given by bids $\boldsymbol{b}$ is a stable assignment, that is, no advertiser can profitably rematch by changing her position with any other advertiser. Thus, $[\![(\text{N}_{-\mathsf{i}}, \sigma_{-\mathsf{i}})(\mathsf{i}, t)\mathbf{X}\text{util}_{\mathsf{i}} \leq (\text{N}, \boldsymbol{\sigma})\mathbf{X}\text{util}_{\mathsf{i}}]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v) = 1$. □

As LEFE is still an equilibrium continuum, Edelman et al. (2007) characterize an equilibrium in which the slot allocation and payments coincide with the ones in the dominant-strategy equilibrium (DSE) of the Vickrey–Clarke–Groves (VCG) mechanism.

Let $\mathbf{v} = (\mathsf{v}_{\mathsf{i}})_{\mathsf{i}\in\text{N}}$ be a valuation profile. Truthfully reporting $\mathbf{v}$ is the DSE of VCG (Nisan et al., 2007). For each slot s and agent i, the allocation rule for VCG in the keyword auction is the same as under GSP (Edelman et al., 2007): $\text{all}^*_{\mathsf{i},s}(\mathbf{v}) = 1$ if $rank_{\mathbf{v}}(s) = \mathsf{i}$ and $s \leq \mathsf{n}$. Otherwise, $\text{all}^*_{\mathsf{i},s}(\mathbf{v}) = 0$. The payment for the last slot m is $\text{pay}^*_{\mathsf{m}}(\mathbf{v}) = \theta_{\mathsf{m}} \cdot \mathsf{v}_{rank_{\mathbf{v}}(\mathsf{m}+1)}$ if $\mathsf{m}+1 \leq \mathsf{n}$ and $\text{pay}^*_{\mathsf{m}} = 0$ otherwise. For the remaining slots $1 \leq s < \mathsf{m}$, $\text{pay}^*_s(\mathbf{v}) = (\theta_s - \theta_{s+1}) \cdot \mathsf{v}_{rank_{\mathbf{v}}(s+1)} + \text{pay}^*_{s+1}(\mathbf{v})$. We assume $\text{pay}^*_s(\mathbf{v})$ and $\text{all}^*_{s,\mathsf{i}}(\mathbf{v})$ are functions in $\mathcal{F}$.

6.3.1.3 VCG outcome

The following formula denotes whether the allocation and payments in the next state are the same as the ones for the VCG when agents bid truthfully:

$$\varphi_{\text{VCG}}(\boldsymbol{\sigma}) =_{\text{def}} (\text{N}, \sigma)\mathbf{X}[\bigwedge_{s\in S}(\text{pay}_s = \text{pay}^*_s(\boldsymbol{\vartheta}) \wedge \bigwedge_{\mathsf{i}\in\text{N}} \text{all}_{s,\mathsf{i}} = \text{all}^*_{s,\mathsf{i}}(\boldsymbol{\vartheta}))]$$

If a strategy profile leads to the VCG outcome, then it is a LEFE:

Proposition 6.2. *For any state $v \in V$, $\rho \in \{iR, ir\}$ and strategy profile $\boldsymbol{\sigma} = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\text{N}}$ with $\sigma_{\mathsf{i}} \in \text{Str}^{\rho}_{\mathsf{i}}$ for each i, $[\![\varphi_{\text{VCG}}(\boldsymbol{\sigma}) \to \text{LEFE}(\boldsymbol{\sigma})]\!]^{\mathcal{G}_{GSP},\rho}(v) = 1$.*

Proof sketch. Let $\mathcal{A}$ be an assignment, $v \in V$ be a state, $\rho \in \{iR, ir\}$ and $\boldsymbol{\sigma} = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\text{N}}$ be a profile of ρ-strategies. We denote $\mathbf{v} = (\mathsf{v}_{\mathsf{i}})_{\mathsf{i}\in\text{N}}$ where $\mathsf{v}_{\mathsf{i}} = \ell(v, \vartheta_{\mathsf{i}})$. Assume $[\![\varphi_{\text{VCG}}(\boldsymbol{\sigma})]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v) = 1$, then we have $[\![(\text{N}, \boldsymbol{\sigma})\mathbf{X}[\bigwedge_{s\in S}(\text{pay}_s = \text{pay}^*_s(\boldsymbol{\vartheta}) \wedge \bigwedge_{\mathsf{i}\in\text{N}} \text{all}_{s,\mathsf{i}} = \text{all}^*_{s,\mathsf{i}}(\boldsymbol{\vartheta}))]]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v) = 1$. We denote by $v_{\boldsymbol{\sigma}} = \delta(v, \boldsymbol{b})$ the state succeeding v when agents follow $\boldsymbol{\sigma}$, where $\boldsymbol{b} = (b_{\mathsf{r}})_{\mathsf{r}\in\text{N}}$ and $b_{\mathsf{r}} = act_{match(v,\sigma_{\mathsf{r}})}$. Let $s > 1$ be a slot and i be an agent. By the definition of φ_{VCG}, $\ell(v_{\boldsymbol{\sigma}}, \text{all}_{s,\mathsf{i}}) = 1$ if $rank_{\mathbf{v}}(s) = \mathsf{i}$ and $s \leq \mathsf{n}$. Otherwise, $\text{all}^*_{\mathsf{i},s} = 0$.

Given that allocations in $v_{\boldsymbol{\sigma}}$ are the same as in the (truthful) outcome of VCG, it must be the case that $rank_{\boldsymbol{b}}(s) = rank_{\mathbf{v}}(s)$. Thus, agents were allocated by descending order of their valuations (recall the valuations are distinct).

According to the weight function, each agent is allocated to at most one slot. We consider first the case in which i is not allocated to any slot, *i.e.*,

$max_{s' \in S}(\ell(v_{\boldsymbol{\sigma}}, \mathrm{all}_{\mathsf{i},s'})) = 0$. This case happens when $\mathsf{m} < \mathsf{n}$, that is, there are no enough slots for all agents. The utility of i for the slot m is $\mathrm{util}_{\mathsf{i},\mathsf{m}} = \theta_{\mathsf{m}}(\mathsf{v}_{\mathsf{i}} - \ell(v_{\boldsymbol{\sigma}}, \mathrm{pay}_{\mathsf{m}}))$. By the definition of $\mathrm{pay}^*_{\mathsf{m}}$, $\mathrm{util}_{\mathsf{i},\mathsf{m}} = \theta_{\mathsf{m}}(\mathsf{v}_{\mathsf{i}} - \theta_{\mathsf{m}}\mathsf{v}_{rank_{\mathbf{v}(\mathsf{m}+1)}})$. That is, $\mathrm{util}_{\mathsf{i},\mathsf{m}} = \theta_{\mathsf{m}}\mathsf{v}_{\mathsf{i}} - \theta^2_{\mathsf{m}}\mathsf{v}_{rank_{\mathbf{v}(\mathsf{m}+1)}}$. Since $\mathsf{v}_{\mathsf{i}} < \mathsf{v}_{rank_{\boldsymbol{b}}(\mathsf{m}+1)}$, we have that $\mathrm{util}_{\mathsf{i},\mathsf{m}} < 0$. Thus, $[\![0 > \mathrm{util}_{\mathsf{i},\mathsf{m}}]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v_{\boldsymbol{\sigma}}) = 1$ and $[\![\mathrm{LEF}_{loses}]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v_{\boldsymbol{\sigma}}) = 1$.

Now we verify the case i was assigned to a slot $1 < s \leq \mathsf{m}$. Assume for the sake of contradiction, that $[\![\mathrm{util}_{\mathsf{i},s-1} > \mathrm{util}_{\mathsf{i},s}]\!]^{\mathcal{G}_{GSP},\rho}_{\mathcal{A}}(v_{\boldsymbol{\sigma}}) = 1$. Then, in the game induced by VCG, i would have an incentive to switch her bid with the agent in slot $s-1$, which is a contradiction since the bidding v_{i} is the dominant strategy for i in VCG. □

In fact, from (Edelman et al., 2007; Varian, 2007) the VCG payments are the lower bound of locally envy-free equilibrium. Thus, in any other locally envy-free equilibrium the total revenue obtained by GSP is at least as high as the one obtained by VCG in equilibrium.

Corollary 6.1. *For any state $v \in V$, $\rho \in \{iR, ir\}$ and strategy profile $\boldsymbol{\sigma} = (\sigma_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}$ with $\sigma_{\mathsf{i}} \in \mathrm{Str}^{\rho}_{\mathsf{i}}$ for each agent i, $[\![\mathrm{LEFE}(\boldsymbol{\sigma}) \rightarrow \sum_{s\in S}(\mathrm{pay}_s) \geq \sum_{s\in S}(\mathrm{pay}^*_s)]\!]^{\mathcal{G}_{GSP},\rho}(v) = 1$.*

The solution concepts characterized in this section are considered in a single stage of the game. Since the auction is repeated, advertisers can change their bids very frequently and one may investigate whether the prices stabilize and at what values (Edelman et al., 2007). Stable bids must be best responses to each other, that is, the bids form an (one-shot) equilibrium. Cary et al. (2007) raises the problem on whether there exists a "natural bidding strategy" for the advertisers that would lead to equilibrium.

Convergence The concept of convergence or stabilization can be easily encoded in $\mathsf{NatSL}[\mathcal{F}]$: we say a wCGSl $\mathcal{G}$ converge to a property φ if the initial states lead to φ being eventually always the case. Formally, a wCGSl converge to a condition φ if $[\![\mathbf{FG}(\varphi)]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v_\iota) = 1$ for each initial state $v_\iota \in V_\iota$.

6.3.2 Natural Strategies for GSP

Given agent i and the wCGSl $\mathcal{G}_{GSP}$, we exemplify strategies for i in a repeated keyword auction. For readability, we omit the epistemic operator K_{i} from an epistemic condition $K_{\mathsf{i}}\varphi$ when the satisfaction value of φ is known by i in all states. A common approach for an advertiser is to assume that all the other bids will remain fixed in the next round and target the slot that maximizes her utility at current prices. This mechanism allows a range of bids that will result in the same outcome from i's perspective, so a number of strategies are distinguished by the bid choice within this range.

6.3.2.1 Balanced bidding

In the balanced bidding strategy (BB) (Cary et al., 2007), the agent bids so as to be indifferent between successfully winning the targeted slot at its current price, or winning a slightly more desirable slot at her bid price. The natural strategy representing balanced bidding for agent i is denoted BB_{i} and is constructed in three parts. First, include the guarded actions $(BB_{\mathsf{i},1}(b), b)$ for each action $b \in \mathcal{B}$. Second, include $(BB_{\mathsf{i},2}(b,s), b)$ for each $b \in \mathcal{B}$ and $1 < s \leq \mathsf{m}$. Third, the last guarded action is $(\top, 0)$. The condition $BB_{\mathsf{i},1}(b)$ refers to the case in which the slot maximizing i's utility is the top slot and b is $(\vartheta_{\mathsf{i}} + \text{pay}_1)/2$:

$$BB_{\mathsf{i},1}(b) =_{\text{def}} b = \frac{\vartheta_{\mathsf{i}} + \text{pay}_1}{2} \wedge \frac{1}{argmax_{s \in S(\text{util}_{\mathsf{i},s})}} = 1$$

Condition $BB_{\mathsf{i},2}(b,s)$ denotes the case in which the slot $s \neq 1$ maximizes i's utility and b is the bid value that is high enough to force the prices paid by her competitors to rise, but not so high that she would mind getting a higher slot at a price just below b.

$$BB_{\mathsf{i},2}(b,s) =_{\text{def}} \text{util}_{\mathsf{i},s} = \theta_{s-1} \times (\vartheta_{\mathsf{i}} - b) \wedge \frac{1}{argmax_{s' \in S}(\text{util}_{\mathsf{i},s'})} = \frac{1}{s}$$

Notice the guarded action $BB_{\mathsf{i},2}(b,s)$ is defined for $s > 1$ since it compares the utility with the one for $s-1$. The case $s = 1$ is treated by the guarded action $BB_{\mathsf{i},1}(b)$.

Given a valuation profile $\mathbf{v} = (\mathsf{v}_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}$, let η_x be the agent in the x-th position of $rank_{\mathbf{v}}$ (that is, η_x is the agent with x-th highest valuation). We let $b_{\eta_x}(\mathbf{v})$ be a function in $\mathcal{F}$ defined as follows:

$$b_{\eta_x}(\mathbf{v}) = \begin{cases} \frac{\theta_x}{\theta_{x-1}} \cdot b_{rank_{\mathbf{v}}(x+1)}(\mathbf{v}) + (1 - \frac{\theta_x}{\theta_{x-1}})\mathsf{v}_{\eta_x} & \text{if } x \geq \mathsf{m}+1 \\ \mathsf{v}_{\eta_x} & \text{if } 2 \leq x \leq \mathsf{m} \end{cases}$$

If $\boldsymbol{BB} = (BB_{\mathsf{i}})_{\mathsf{i} \in \mathrm{N}}$ converges to the equilibrium with VCG outcomes, the agent with the highest valuation, that is η_1, bids any value above $b_{\eta_2}(\mathbf{v})$. The equilibrium bid for $\mathsf{i} \neq \eta_1$ is $b_{\mathsf{i}}(\mathbf{v})$ (Cary et al., 2007). When there are two slots and all players update their bids according to BB, the game converges to the equilibrium with VCG outcome. However, this is not the case for more than two slots (Cary et al., 2007).

Proposition 6.3. *For any initial state $v_\iota \in V_\iota$, state $v \in V$, and $1 < x \leq \mathsf{n}$, the following holds, where $\mathbf{v} = (\ell(v, \vartheta_{\mathsf{i}}))_{\mathsf{i} \in \mathrm{N}}$:*

1. *If $[\![\varphi_{\text{VCG}}(\boldsymbol{BB})]\!]^{\mathcal{G}_{GSP},ir}(v) = 1$, then $act_{match(v,BB_{\eta_x})} = b_{\eta_x}(\mathbf{v})$ and $act_{match(v,BB_{\eta_1})} > b_{\eta_2}(\mathbf{v})$;*
2. *If $\mathsf{m} = 2$, then $[\![\mathbf{FG}(\varphi_{\text{VCG}}(\boldsymbol{BB}))]\!]^{\mathcal{G}_{GSP},ir}(v_\iota) = 1$;*

3. If $\mathsf{m} \geq 3$*, then* $[\![\mathbf{FG}(\varphi_{\mathrm{VCG}}(\boldsymbol{BB}))]\!]^{\mathcal{G}_{GSP},ir}(v_\iota) \neq 1$.

Proof sketch. Statement (1) is derived in (Cary et al., 2007) from the results of Edelman et al. (2007). Notice that when each agent i is bound to the natural strategy BB_i, they will update their bids simultaneously in every state reachable form v. Thus, it corresponds to the synchronous setting described by Cary et al. (2007). The proof for Statements (2) and (3) are very similar to the one provided in the analysis of the synchronous setting by Cary et al. (2007). □

6.3.2.2 Restricted BB

The restricted balanced bidding strategy (RBB) (Cary et al., 2007) is a variation of BB in which the agent only targets slots that are not better than her current slot. The natural strategy representing RBB for agent i is denoted RBB_i and is constructed as follows. First, include the guarded actions $(RBB_{\mathsf{i},1}(b), b)$ for each action $b \in \mathcal{B}$. Second, include $(RBB_{\mathsf{i},2}(b,s), b)$ for each $b \in \mathcal{B}$ and $1 < s \leq \mathsf{m}$. Finally, the last guarded action is $(\top, 0)$. Let $s_\mathsf{i} = min(\mathsf{m}, \sum_{s' \in S} s' \times \mathrm{all}_{\mathsf{i},s'})$ be the slot assigned to agent i or the last slot if there is no such slot. We define $RBB_{\mathsf{i},1}(b)$ and $RBB_{\mathsf{i},2}(b,s)$:

$$RBB_{\mathsf{i},1}(b) =_{\mathrm{def}} b = \frac{\vartheta_\mathsf{i} + \mathrm{pay}_1}{2} \wedge \frac{1}{argmax_{s \in S \& s \geq s_\mathsf{i}}(\mathrm{util}_{\mathsf{i},s})} = 1$$

$$RBB_{\mathsf{i},2}(b,s) =_{\mathrm{def}} \mathrm{util}_{\mathsf{i},s} = \theta_{s-1} \times (\vartheta_\mathsf{i} - b) \wedge \frac{1}{argmax_{s' \in S \& s' \geq s_\mathsf{i}}(\mathrm{util}_{\mathsf{i},s'})} = \frac{1}{s}$$

Similar to the results in (Cary et al., 2007), we have that if all agents follow the restricted balanced-bidding strategy, the auction converge to the VCG equilibrium outcome. RBB always converge:

Proposition 6.4. *For any initial state* $v_\iota \in V_\iota$*, state* $v \in V$*, and* $1 < x \leq \mathsf{n}$*, the following holds, where* $\mathbf{v} = (\ell(v, \vartheta_\mathsf{i}))_{\mathsf{i} \in \mathrm{N}}$*:*

1. *If* $[\![\varphi_{\mathrm{VCG}}(\boldsymbol{RBB})]\!]^{\mathcal{G}_{GSP},ir}(v) = 1$*, then* $act_{match(v,RBB_{\eta_x})} = b_{\eta_x}(\mathbf{v})$ *and* $act_{match(v,RBB_{\eta_1})} > b_{\eta_2}(\mathbf{v})$*;*
2. $[\![\mathbf{FG}(\varphi_{\mathrm{VCG}}(\boldsymbol{RBB}))]\!]^{\mathcal{G}_{GSP},ir}(v_\iota) = 1$.

Proof sketch. Statement (1) is a derivation from the results presented in (Edelman et al., 2007). For Statement (2), the proof is similar to the one provided in (Cary et al., 2007). The proof idea is the following. First bound the number of steps until convergence of the price of slot m and the set of players who will not be allocated slots. After this step, no losing player can afford a slot and their bids do not interfere with the convergence of the top m agents. The second stage of the proof is to show that the allocation of the top m players converges to a fixed point (in which they are sorted by their valuations). Then, for $1 \leq i \leq \mathsf{m}$, the proof inductively considers

the allocation of slots $[i+1, \mathsf{m}]$. A subset of slots is called stable if the allocation is in order of decreasing values and if agent η_j is the player currently allocated slot j, then her last bid is in accordance with $b_{\eta_j}(\boldsymbol{\vartheta})$ for every $j \in [i+1, \mathsf{m}]$. While the current setting is not a fixed point of RBB, the proof proceeds by characterizing the number of rounds taken for increasing the size of the maximal stable set. □

6.3.2.3 Knowledge grounded RBB

The knowledge grounded RBB strategy (KBB) is a variation of RBB in which the agent uses her knowledge about the valuation of the player currently at her target slot to ground her bid value. The idea is to avoid bidding more than what she knows her opponent valuates the slot. The natural strategy representing KBB for agent i is denoted KBB_{i} is constructed in three steps. First, include the guarded actions $(KBB_{\mathsf{i},1}(b, c, \mathsf{r}), c)$ for each $b, c \in \mathcal{B}$ and agent $\mathsf{r} \neq \mathsf{i}$. Second, include $(KBB_{\mathsf{i},2}(b, s, c, \mathsf{r}), c)$ for each $b, c \in \mathcal{B}$, slot $1 < s \leq \mathsf{m}$ and agent $\mathsf{r} \neq \mathsf{i}$. Finally, include the guarded actions from RBB_{i}. The conditions $KBB_{\mathsf{i},1}(b, c, \mathsf{r})$ and $KBB_{\mathsf{i},2}(b, s, c, \mathsf{r})$ are defined as follows:

$$KBB_{\mathsf{i},1}(b, c, \mathsf{r}) =_{\text{def}} K_{\mathsf{i}}\big(RBB_{\mathsf{i},1}(b) \wedge \text{all}_{\mathsf{r},1} = 1 \wedge c = min(\vartheta_{\mathsf{r}}, b)\big)$$

$$KBB_{\mathsf{i},2}(b, s, c, \mathsf{r}) =_{\text{def}} K_{\mathsf{i}}\big(RBB_{\mathsf{i},2}(b, s) \wedge \text{all}_{\mathsf{r},s} = 1 \wedge c = min(\vartheta_{\mathsf{r}}, b)\big)$$

The prices under KBB are at most the same as under RBB:

Proposition 6.5. *For any state $v \in V$, slot $s \in S$ and agent $\mathsf{i} \in \mathrm{N}$, $[\![(\mathrm{N}, \boldsymbol{KBB})\text{pay}_s \leq (\mathrm{N}, \boldsymbol{RBB})\text{pay}_s]\!]^{\mathcal{G}_{GSP}, ir}(v) = 1$.*

Proof. Consequence from the construction of $\boldsymbol{KBB}$. □

Remark 6.1. With natural strategies, we can easily construct a strategy in which agent η_x plays according to $b_{\eta_x}(\mathbf{v})$ (for $1 < x \leq \mathsf{n}$) and agent η_1 bids $b_{\eta_2} + \mathsf{inc}$ when she knows others' valuations.

BB with recall Since BB may not converge to the VCG equilibrium outcome due to loops on the slot allocation and prices, we construct a strategy that behaves according to BB while there is no repetition in the outcome and follows RBB otherwise. Hereafter, we show that this strategy with recall prevents the loops that hinder the convergence of BB. Define the set of weighted conditions $\Psi = \{\bigwedge_{s \in S}(\text{pay}_s = pr_s \wedge \bigwedge_{\mathsf{i} \in \mathrm{N}} \text{all}_{\mathsf{i},s} = al_{\mathsf{i},s}) : pr_s \in \mathcal{B} \;\&\; al_{\mathsf{i},s} \in \{0, 1\}\}$. The natural strategy representing balanced bidding with recall for agent i is denoted BBR_{i} and is constructed as follows. First, include the guarded actions $(BBR_{\mathsf{i},1}(\psi, b), b)$ for each action $b \in \mathcal{B}$ and condition $\psi \in \Psi$. Second, include $(BBR_{\mathsf{i},2}(\psi, b, s), b)$ for each $\psi \in \Psi$, $b \in \mathcal{B}$ and $1 < s \leq \mathsf{m}$. Third, include $(BBR_{\mathsf{i},3}(\psi, b), b)$ for each action $b \in \mathcal{B}$. Fourth, include $(BBR_{\mathsf{i},4}(b, s), b)$ for each $b \in \mathcal{B}$ and $1 < s \leq \mathsf{m}$. Finally, the last guarded action is $(\top^*, 0)$.

Now we define each guarded condition in BBR_{i}. If the current allocation and payments have already happen in the past, i plays according to the restricted bidding strategy:

$$BBR_{\mathsf{i},1}(\psi, b) =_{\text{def}} \top^* \cdot \psi \cdot \top^* \cdot (\psi \wedge RBB_{\mathsf{i},1}(b))$$

$$BBR_{\mathsf{i},2}(\psi, b, s) =_{\text{def}} \top^* \cdot \psi \cdot \top^* \cdot (\psi \wedge RBB_{\mathsf{i},2}(b, s))$$

If there was no repetition on the payments and slot allocation, she plays according to the balanced bidding strategy:

$$BBR_{\mathsf{i},3}(\psi, b) =_{\text{def}} \top^* \cdot BB_{\mathsf{i},1}(b)$$

$$BBR_{\mathsf{i},4}(\psi, b, s) =_{\text{def}} \top^* \cdot BB_{\mathsf{i},2}(b, s)$$

When all agents follow the strategy profile $\boldsymbol{BBR} = (BBR_{\mathsf{i}})_{\mathsf{i}\in \mathrm{N}}$, the game converges to the VCG equilibrium outcome.

Proposition 6.6. *For any initial state $v_\iota \in V_\iota$, state $v \in V$, and $1 < x \leq \mathsf{n}$, the following holds, where $\mathbf{v} = (\ell(v, \vartheta_{\mathsf{i}}))_{\mathsf{i}\in \mathrm{N}}$:*

1. *If $[\![\varphi_{\mathrm{VCG}}(\boldsymbol{BBR})]\!]^{\mathcal{G}_{GSP}, iR}(v) = 1$, then $act_{match(v, BBR_{\eta_x})} = b_{\eta_x}(\mathbf{v})$ and $act_{match(v, BBR_{\eta_1})} > b_{\eta_2}(\mathbf{v})$;*
2. *$[\![\mathbf{FG}(\varphi_{\mathrm{VCG}}(\boldsymbol{BBR}))]\!]^{\mathcal{G}_{GSP}, iR}(v_\iota) = 1$.*

Proof sketch. Statement (1) follows from Propositions 6.3 and 6.4. Statement (2) is proven by contradiction. Assume the game does not converge to the VCG equilibrium outcome. Let $\mathcal{A}$ be an assignment and $\pi = \mathrm{Out}(v_\iota, \mathcal{A})$ be the play starting in v_ι and follows the strategies assigned by $\mathcal{A}$. Since $\mathcal{G}_{GSP}$ has finitely many states, there exist two indices $g < l$ such that $\pi_g = \pi_l$. Thus, for every $\psi \in \Psi$, $[\![\psi]\!]^{\mathcal{G}_{GSP}, iR}(\pi_g) = [\![\psi]\!]^{\mathcal{G}_{GSP}, iR}(\pi_l)$. Then, the game proceeds according to RBB strategy. That is, for any index $j \geq l$ and agent i, $act_{match(\pi_l, BBR_{\mathsf{i}})} = act_{match(\pi_l, RBB_{\mathsf{i}})}$. From Proposition 6.4, it follows that $[\![\mathbf{FG}(\varphi_{\mathrm{VCG}}(\boldsymbol{BBR}))]\!]^{\mathcal{G}_{GSP}, iR}_{\mathcal{A}}(v_\iota) = 1$. □

When other agents are inactive (*i.e.*, they repeat their last action), if BBR_{i} selects a different bid from the one assigned by RBB_{i}, the utility of i in the next state is greater under BBR_{i}.

Proposition 6.7. *Let $\rho \in \{ir, iR\}$ and $v = \delta(v', \boldsymbol{a})$, for some state $v' \in V$ and action profile $\boldsymbol{a} = (a)_{\mathsf{i}\in \mathrm{N}}$. Given an agent $\mathsf{i} \in \mathrm{N}$, let $\sigma^\rho_{-\mathsf{i}} = (\sigma^\rho_{\mathsf{r}})_{\mathsf{r}\in \mathrm{N}_{-\mathsf{i}}}$ be a ρ-strategy profile, where the strategy σ^ρ_{r} of agent r is such that $act_{match(v, \sigma^\rho_{\mathsf{r}})} = a_{\mathsf{r}}$. If $act_{match(v, BBR_{\mathsf{i}})} \neq act_{match(v, RBB_{\mathsf{i}})}$, then $[\![(\mathrm{N}_{-\mathsf{i}}, \sigma^{iR}_{-\mathsf{i}})(\mathsf{i}, BBR_{\mathsf{i}})\mathbf{X}\mathrm{util}_{\mathsf{i}}]\!]^{\mathcal{G}_{GSP}, iR}(v) > [\![(\mathrm{N}_{-\mathsf{i}}, \sigma^{ir}_{-\mathsf{i}})(\mathsf{i}, RBB_{\mathsf{i}})\mathbf{X}\mathrm{util}_{\mathsf{i}}]\!]^{\mathcal{G}_{GSP}, ir}(v)$.*

Proof sketch. Assume the actions assigned by BBR_{i} and RBB_{i} are different, that is, $act_{match(v, BBR_{\mathsf{i}})} \neq act_{match(v, RBB_{\mathsf{i}})}$, then it must be the case that $act_{match(v, BBR_{\mathsf{i}})} = b$ such that $[\![\mathrm{util}_{\mathsf{i},s} = \theta_{s-1} \times (\vartheta_{\mathsf{i}} - b) \wedge (argmax_{s'\in S}(\mathrm{util}_{\mathsf{i},s'}))^{-1} = s^{-1}]\!]^{\mathcal{G}_{GSP}, ir}_{\mathcal{A}}(v) = 1$ for some slot $s > 1$. Notice that RBB_{i} selected the action $act_{match(v, RBB_{\mathsf{i}})}$ that

maximizes the utility among slots that are better or equal to i's current slot. By the other hand, the condition followed by BBR_{i} in v chose the action b that maximizes among all slots. Thus, since the actions are different, b has the greatest estimated utility assuming the others repeat their previous bids. As it is in fact the case, by the definition of $\sigma^{iR}_{-\mathsf{i}}$ and $\sigma^{ir}_{-\mathsf{i}}$, i's is assigned to s in the next state and her utility is the one estimated, that is $[\![\mathbf{X}\mathrm{util}_{\mathsf{i}}]\!]^{\mathcal{G}_{GSP},iR}_{\mathcal{A}}(v) = [\![\mathrm{util}_{\mathsf{i},s}]\!]^{\mathcal{G}_{GSP},iR}_{\mathcal{A}}(v)$. Thus, $[\![(\mathrm{N}_{-\mathsf{i}}, \sigma^{iR}_{-\mathsf{i}})(\mathsf{i}, BBR_{\mathsf{i}})\mathbf{X}\mathrm{util}_{\mathsf{i}}]\!]^{\mathcal{G}_{GSP},iR}(v) > [\![(\mathrm{N}_{-\mathsf{i}}, \sigma^{ir}_{-\mathsf{i}})(\mathsf{i}, RBB_{\mathsf{i}})\mathbf{X}\mathrm{util}_{\mathsf{i}}]\!]^{\mathcal{G}_{GSP},ir}_{\mathcal{A}}(v)$. □

Remark 6.2. In vindictive bidding (Zhou and Lukose, 2007), the agent bids as high as possible to raise the payment of the advisor in the slot right below hers. Since there is the risk that a change in other agents' bids could result in paying a higher price than expected, the player could use memory to balance the use of aggressive bids. Strategies with recall could also be used for managing budget. For strategies on budget-constrained keyword auctions the reader may refer to Zhou et al. (2008).

In the next sections, we move our focus to the analysis of $\mathsf{NatSL}[\mathcal{F}]$ and we provide technical results in terms of expressivity and model-checking complexity.

6.4 Expressivity

In relation to $\mathsf{SL}[\mathcal{F}]$ with combinatorial strategies, $\mathsf{NatSL}[\mathcal{F}]$ introduces a new, broader class of human-friendly strategies *and* a language for expressing properties of agents that use such strategies. Clearly, strategies with quantitative conditions can be used to obtain goals that would not be achievable otherwise. On the other hand, bounded natural strategies of $\mathsf{NatSL}[\mathcal{F}]$ may not achieve some goals that can be enforced with combinatorial strategies of $\mathsf{SL}[\mathcal{F}]$. In this section, we show that the expressive power of $\mathsf{NatSL}[\mathcal{F}]$ is incomparable to that of $\mathsf{SL}[\mathcal{F}]$. In other words, there are properties of quantitative games with natural strategies that cannot be equivalently translated to properties based on combinatorial strategies, and vice versa. From this, we conclude that reasoning about human-friendly strategies offers an inherently different view of a multi-agent system from the "standard" one.

6.4.1 Expressive and Distinguishing Power

We first adapt the notions of distinguishing power and expressive power to the quantitative case as follows[4].

Definition 6.4 (Distinguishing power of real-valued logics)**.** Let $\mathcal{L}_1 = (L_1, [\![\cdot]\!]_1)$ and $\mathcal{L}_2 = (L_2, [\![\cdot]\!]_2)$ be two logical systems with syntax L_1, L_2 and real-valued semantics $[\![\cdot]\!]_1, [\![\cdot]\!]_2$ over the same class of models $\mathcal{M}$. We say that $\mathcal{L}_2$ is *at least as distinguishing* as $\mathcal{L}_1$ (written: $\mathcal{L}_1 \preceq_d \mathcal{L}_2$) iff for every pair of models $M, M' \in \mathcal{M}$, if there exists a formula $\varphi_1 \in L_1$ such that $[\![\varphi_1]\!]^M_1 \neq [\![\varphi_1]\!]^{M'}_1$, then there is also $\varphi_2 \in L_2$

[4] *Cf.,e.g.,* (Wang and Dechesne, 2009) for a detailed discussion of standard notions of expressivity.

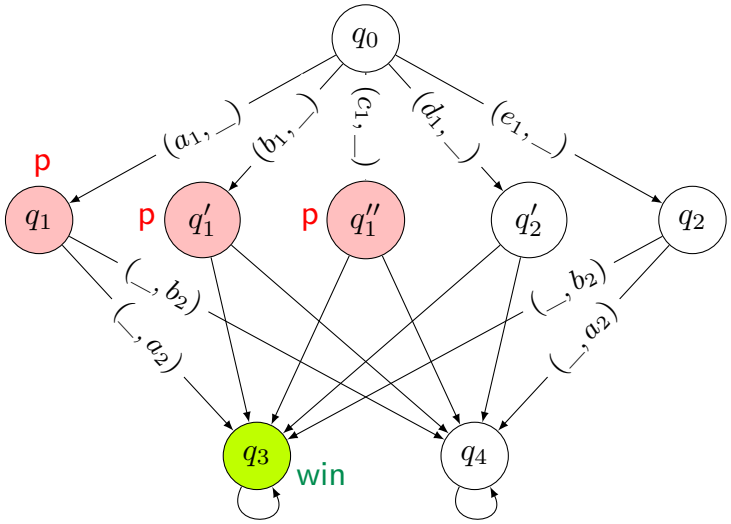

Figure 6.1: Model $\mathcal{G}_1$. Its counterpart $\mathcal{G}_1'$ is obtained by fixing p to hold only in q_1, q_1'. Underscore fits any action label

with $[\![\varphi_2]\!]_2^M \neq [\![\varphi_2]\!]_2^{M'}$. In other words, if there is a formula of $\mathcal{L}_1$ discerning M from M', then there must be also a formula of $\mathcal{L}_2$ doing the same.

Definition 6.5 (Expressive power of real-valued logics)**.** $\mathcal{L}_2$ is *at least as expressive* as $\mathcal{L}_1$ (written: $\mathcal{L}_1 \preceq_e \mathcal{L}_2$) iff for every $\varphi_1 \in L_1$ there exists $\varphi_2 \in L_2$ such that, for every model $M \in \mathcal{M}$, we have $[\![\varphi_1]\!]_1^M = [\![\varphi_2]\!]_2^M$. In other words, every formula of $\mathcal{L}_1$ has a translation in $\mathcal{L}_2$ that produces exactly the same truth values on models in $\mathcal{M}$.

It is easy to see that $\mathcal{L}_1 \preceq_e \mathcal{L}_2$ implies $\mathcal{L}_1 \preceq_d \mathcal{L}_2$. Thus, by transposition, we also get that $\mathcal{L}_1 \not\preceq_d \mathcal{L}_2$ implies $\mathcal{L}_1 \not\preceq_e \mathcal{L}_2$.

In the remainder, $\mathcal{M}$ is the class of pointed weighted games, i.e., pairs $(\mathcal{G}, v)$ where $\mathcal{G}$ is a wCGSl and v is a state in $\mathcal{G}$.

6.4.2 Expressivity of NatSL[$\mathcal{F}$] vs. SL[$\mathcal{F}$]

NatSL[$\mathcal{F}$] and SL[$\mathcal{F}$] are based on different notions of strategic ability. The former refers to "natural" strategies, represented as mappings from regular expressions over epistemic formulas to actions. The latter uses "combinatorial" strategies, represented by mappings from sequences of states to actions. Each natural strategy can be translated to a combinatorial one, but not vice versa. Consequently, SL[$\mathcal{F}$] can express that a given coalition has a combinatorial strategy to achieve their goal (which is not expressible in NatSL[$\mathcal{F}$]). On the other hand, NatSL[$\mathcal{F}$] allows expressing that a winning natural strategy does not exist (which cannot be captured in SL[$\mathcal{F}$]). Now we show that NatSL[$\mathcal{F}$] allows to express properties of MAS that cannot be captured in SL[$\mathcal{F}$], and vice versa.

Proposition 6.8. NatSL[$\mathcal{F}$] $\not\preceq_d$ SL[$\mathcal{F}$] *in both ir and iR semantics.*

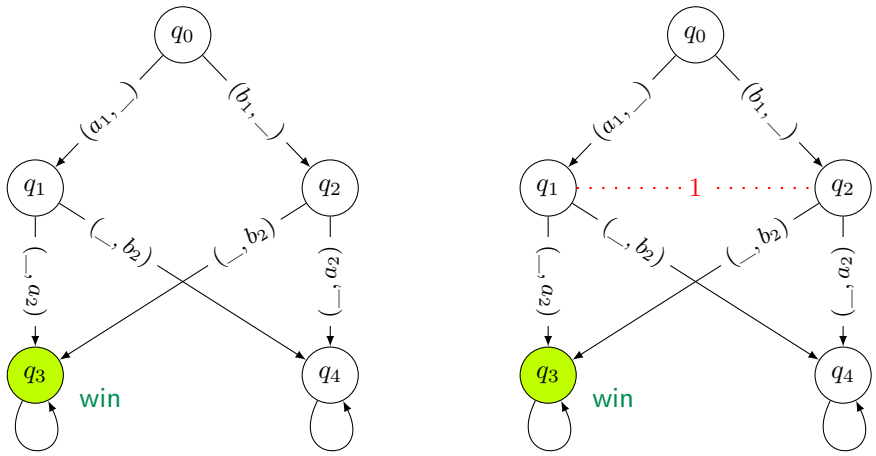

Figure 6.2: Models $\mathcal{G}_2$ (left) and $\mathcal{G}'_2$ (right)

Proof sketch. Consider model $\mathcal{G}_1$ in Figure 6.1, with agents $\mathrm{N} = \{1, 2\}$, actions $\mathcal{B}_1 = \{a_1, b_1, c_1, d_1, e_1\}$ and $\mathcal{B}_2 = \{a_2, b_2\}$ available at all positions, and propositions $\mathrm{AP} = \{\mathsf{p}, \mathsf{win}\}$. Both propositions are qualitative (that is, the propositions have only values in {-1,1}). For each proposition, the states where it evaluates to 1 are indicated; otherwise its truth value is assumed to be -1. The outgoing transitions in q'_1, q''_1 (resp. q'_2) are exact copies of those at q_1 (resp. q_2). Moreover, model $\mathcal{G}'_1$ is obtained by fixing proposition p to hold only in q_1, q'_1, but not in q''_1. As all the propositions are qualitative, formulas of $\mathsf{NatSL}[\mathcal{F}]$ and $\mathsf{SL}[\mathcal{F}]$ evaluate to -1 or 1. Note also that the sets of *ir* and *iR* strategies in each model coincide, so we can concentrate on the *ir* case w.l.o.g.

Let $\mathcal{G}\dagger\sigma$ denote the model obtained by fixing the (memoryless) strategy σ in $\mathcal{G}$. In order to prove that $(\mathcal{G}_1, q_0)$ and $(\mathcal{G}'_1, q_0)$ satisfy the same formulas of $\mathsf{SL}[\mathcal{F}]$, it suffices to observe that:

1. For every strategy σ_1 of agent 1 in $\mathcal{G}_1$, there is σ'_1 in $\mathcal{G}'_1$ such that agent 2 has the same strategic abilities in $(\mathcal{G}'_1\dagger\sigma_1, q_0)$ and $(\mathcal{G}'_1\dagger\sigma_1, q_0)$ (and vice versa). For instance, playing c_1 in $\mathcal{G}$ obtains the same abilities of 1 as playing a_1 in $\mathcal{G}'$.

2. Analogously for strategies of agent 2, e.g., strategy $a_2a_2a_2b_2b_2$ in $\mathcal{G}_1$ can be simulated by strategy $a_2a_2b_2b_2b_2$ in $\mathcal{G}'_1$.

On the other hand, the formula $\exists s_2^{\leq 2} \forall s_1^{\leq 1} (1, s_1)(2, s_2)\mathbf{F}\mathsf{win}$ of $\mathsf{NatSL}[\mathcal{F}]$ holds in $(\mathcal{G}_1, q_0)$, but not in $(\mathcal{G}'_1, q_0)$. The winning natural strategy for agent 2 in $\mathcal{G}_1$ is $((\top^*\mathsf{p}, a_2),\ (\top^*, b_2))$; clearly, it does not succeed in $\mathcal{G}'_1$. □

Proposition 6.9. $\mathsf{SL}[\mathcal{F}] \not\equiv_d \mathsf{NatSL}[\mathcal{F}]$ *in both ir and iR semantics.*

Proof sketch. Consider models $\mathcal{G}_2$ and $\mathcal{G}'_2$ in Figure 6.2. They have isomorphic action/transition structures, the only difference being the indistinguishability of states q_1, q_2 in $\mathcal{G}'_2$ (but not in $\mathcal{G}_2$). Since the two states have the same valuations of

propositions, each natural strategy must specify the same decision in q_1, q_2. Thus, both players have exactly the same available natural strategies in $\mathcal{G}_2$ and $\mathcal{G}'_2$, and hence $(\mathcal{G}_2, q_0)$ and $(\mathcal{G}'_2, q_0)$ produce the same valuations of $\mathsf{NatSL}[\mathcal{F}]$ formulas.

On the other hand, we have that $\exists s_2 \forall s_1 (1, s_1)(2, s_2)\mathbf{F}\mathsf{win}$ of $\mathsf{SL}[\mathcal{F}]$ holds in $(\mathcal{G}_2, q_0)$, but not in $(\mathcal{G}'_2, q_0)$. □

The following is an immediate consequence.

Theorem 6.1. $\mathsf{NatSL}[\mathcal{F}]$ *and* $\mathsf{SL}[\mathcal{F}]$ *have incomparable distinguishing power over the class of pointed wCGSl (in both ir and iR semantics).*

Corollary 6.2. $\mathsf{NatSL}[\mathcal{F}]$ *and* $\mathsf{SL}[\mathcal{F}]$ *have incomparable expressive power over the class of pointed wCGSl (in both ir and iR semantics).*

6.5 Model Checking

In this section we show that the model checking problem for $\mathsf{NatSL}[\mathcal{F}]$ with imperfect information is no harder than model checking LTL or classic SL with memoryless agents. First of all, we define formally the quantitative model-checking problem for $\mathsf{NatSL}[\mathcal{F}]$.

Definition 6.6. Given $\rho \in \{ir, iR\}$, the model-checking problem for $\mathsf{NatSL}[\mathcal{F}]$ consists in deciding, for a given sentence φ, wCGSl $\mathcal{G}$, state $v \in V$ and predicate $P \subseteq [-1; 1]$, whether $[\![\varphi]\!]^{\mathcal{G},\rho}(v) \in P$.

Now, we have all the ingredients to prove the following result.

Theorem 6.2. *Assuming that functions in $\mathcal{F}$ can be computed in polynomial space, model checking* $\mathsf{NatSL}[\mathcal{F}]$ *with imperfect information, natural strategies with recall, and k as parameter of the problem is* PSPACE*-complete.*

Proof. For the lower-bound we recall that $\mathsf{LTL}[\mathcal{F}]$ model checking is PSPACE-complete (Almagor et al., 2016). For the upper-bound, to verify that a given $\mathsf{NatSL}[\mathcal{F}]$ formula φ is satisfied over a wCGSl $\mathcal{G}$ at a state $v \in V$ under assignments $\mathcal{A}$ over uniform natural strategies with recall, we make use of a recursive function as is done in (Cermák et al., 2018). We start by showing that each recursive call only needs at most polynomial space. First, observe that each assignment $\mathcal{A}$ has a strategy s_a for each agent $a \in Ag$[5]. We know that each strategy s_a that can be assigned to agent a is bounded, and we have that $compl(s_a) \leq k$. Thus, each strategy can be stored in $O(k \cdot |Act|)$ and, by consequence, any assignment can be stored in space $O((|Ag| \cdot |\text{free}(\varphi)|) \cdot (k \cdot |Act|))$.

Now, we can analyse the recursive function. For the base case, $[\![p]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v)$ can be computed in constant space via the weight function. For strategy quantification $[\![\exists s_{\mathsf{i}}^{\leq k}. \varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v)$, besides the recursive call to $[\![\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}[s \mapsto \sigma]}(v)$ we need space

[5]Note that, as defined in Section 6.2, we consider only complete assignments. Thus, we can assume that a strategy is assigned for each agent.

$O(|k| \cdot |\mathcal{B}|)$ to store the current strategy and the current maximum value computed. For $[\![f(\varphi_1, \ldots, \varphi_m)]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v)$, by assumption f is computed in polynomial space. For $[\![\mathbf{X}\varphi]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v)$, we only need to observe that the next state in $\mathrm{Out}(v, \mathcal{A})$ is computed in constant space.

Finally, we detail how $[\![\varphi_1 \mathbf{U} \varphi_2]\!]^{\mathcal{G},\rho}_{\mathcal{A}}(v)$ is computed. Let $\pi = \mathrm{Out}(v, \mathcal{A})$. Since $\mathcal{G}$ has finitely many states, there exist two indices $g < l$ such that $\pi_g = \pi_l$, and since strategies are bounded by k, the suffix of π starting at index l is equal to the suffix starting at index g. So there exist $\rho_1 = v_0 \ldots v_{g-1}$ and $\rho_2 = v_g \ldots v_{l-1}$ such that $\pi = \rho_1 \cdot \rho_2^\omega$. It follows that

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

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

Finally, 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. □

Since memoryless natural strategies are a special case of natural strategies with recall, we obtain the following result.

Corollary 6.3. *Assuming that functions in $\mathcal{F}$ can be computed in polynomial space, model checking* $\mathsf{NatSL}[\mathcal{F}]$ *with imperfect information, memoryless natural strategies, and k as parameter of the problem is* PSPACE*-complete.*

6.6 Conclusion

In this chapter we have introduced Natural Strategy Logic with quantitative semantics and imperfect information ($\mathsf{NatSL}[\mathcal{F}]$) for reasoning about strategic ability in auctions. Natural strategies only require remembering a limited number of significant situations. Instead of specifying instructions for every history, they provide instructions for conditions. $\mathsf{NatSL}[\mathcal{F}]$ provides a tool for mechanism design and offers a new perspective for formal verification and design of novel mechanisms and strategies.

Memoryless strategies are enough for several types of mechanisms, as all relevant information is encoded in the current state (for instance, an English auction). In repeated auctions, agents may, as well, use information from the previous instances of the game for choosing their strategies. One relevant problem in repeated auctions is whether the bid prices are going to stabilize and at which price (that is, convergence to an equilibrium). We demonstrated the usefulness of our approach

by modelling and evaluating strategies for repeated keyword auctions. $\mathsf{NatSL}[\mathcal{F}]$ allows for the verification of these strategies in order to provide formal guarantees of correctness as well as improve trust in automated bidders.

In terms of technical results, we proved that the model checking problem for $\mathsf{NatSL}[\mathcal{F}]$ is PSPACE-complete, that is, no harder than model checking for the much less expressive language of quantitative LTL ($\mathsf{LTL}[\mathcal{F}]$). We also showed that $\mathsf{NatSL}[\mathcal{F}]$ has incomparable distinguishing and expressive power to $\mathsf{SL}[\mathcal{F}]$. This means that the characterizations based on simple bounded strategies offer an inherently different view of auctions and mechanism design from characterizations using combinatorial strategies of arbitrary complexity. Surprisingly, this aspect has never been studied for natural strategies, not even for the original proposal of NatATL (Jamroga et al., 2019a).

Up to now, we considered logic-based approaches for the formal verification of mechanisms when faced with strategic agents. In the next chapter, we go further on the AMD problem, by addressing the automated *construction* of mechanisms from logical specifications.

Chapter 7

Synthesis of Mechanisms

In the spirit of the long-established logical approach to systems verification (Clarke et al., 2018) and synthesis (David and Kroening, 2017), in this chapter we propose a new approach to Automated Mechanism Design (AMD). More precisely, we show how AMD 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 solve the problem in two cases: when the number of actions is bounded, and when agents play in turn.

We consider Quantitative Strategy Logic ($\mathsf{SL}[\mathcal{F}]$) (Bouyer et al., 2019), which allows for specifications that contain constraints on mechanism properties (for instance, the efficiency and budget-balance). Its quantitative semantics, with satisfaction values that reflect how well a model satisfies a formula, also allows us to investigate the constructions of mechanisms that approximate such properties, which is not possible with standard Strategy Logic (SL) (Chatterjee et al., 2010; Mogavero et al., 2014).

Unlike Chapter 5, in which we consider strategies without memory, we consider strategies with perfect recall, as in Bouyer et al. (2019). The main reason is that for memoryless strategies satisfiability, and thus synthesis, is undecidable already for SL, even for bounded actions or turn-based systems Laroussinie and Markey (2015). Considering perfect recall strategies means that the systems we synthesize satisfy the strategic aspects of the specification assuming that agents' actions can depend on the past. For similar reasoning, we restrict our attention to wCGS with perfect information (in opposition to the imperfect information setting considered at Chapters 5-6).

7.1 Quantitative Strategy Logic

Let us first recall $\mathsf{SL}[\mathcal{F}]$ (Bouyer et al., 2019) syntax and semantics. As in the previous chapters, we slightly generalize the setting to consider values in $[-1, 1]$.

Definition 7.1. The syntax of $\mathsf{SL}[\mathcal{F}]$ is defined by the grammar

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

where $p \in \mathrm{AP}$, $s \in \mathrm{Var}$, $\mathsf{i} \in \mathrm{N}$, and $f \in \mathcal{F}$.

$\mathsf{SL}[\mathcal{F}]$ syntax and intuition are similar to $\mathsf{SLK}[\mathcal{F}]$, but it does not contain epistemic operators. Furthermore, $\mathsf{SL}[\mathcal{F}]$ does not specify for which agent a strategy is

at the level of strategy quantification, which allows assigning the same strategy to different agents. Thus, $\exists s.\, \varphi$ means that there exists a strategy s such that φ holds.

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

Definition 7.2. A *weighted concurrent game structure* (wCGS) is a tuple $\mathcal{G} = (\mathcal{B}, V, v_\iota, \delta, \ell)$ where each component is defined exactly as the homonyms components in wCGSii (see Definition 5.2)[1].

Remark 7.1. Because we are interested in synthesizing mechanisms of finite size, we restrict attention to finite models. Studying satisfiability of $\mathsf{SL}[\mathcal{F}]$ for potentially infinite models would be of interest too as SL, and thus also $\mathsf{SL}[\mathcal{F}]$, do not enjoy the finite-model property (indeed, SL is more expressive than ATL with strategy context, which does not have the finite-model property; see Laroussinie and Markey (2015)). We leave this question for future work.

Action profiles, plays and assignments for $\mathsf{SL}[\mathcal{F}]$ are defined analogously to the definitions from Chapter 5. By its turn, strategies are now defined over histories.

Histories A *history* h is a finite prefix of a play, $\text{last}(h)$ is the last position of history h, $|h|$ is the length of h and Hist is the set of histories.

Strategies A (perfect recall) *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.

Outcomes 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 $\mathcal{A}$. Formally, $\text{Out}(\mathcal{A}, h)$ is the play $hv_0v_1\ldots$ such that for all $i \geq 0$, $v_i = \delta(v_{i-1}, \boldsymbol{a})$ where for all $\mathsf{i} \in \text{N}$, $\boldsymbol{a}_\mathsf{i} = \mathcal{A}(\mathsf{i})(hv_0 \ldots v_{i-1})$, and $v_{-1} = \text{last}(h)$.

Definition 7.3. 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 $\text{Out}(\mathcal{A}, h)$:

$$
\begin{aligned}
[\![p]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= \ell(\text{last}(h), p) \\
[\![\exists s.\, \varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= \max_{\sigma \in Str} [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[s \mapsto \sigma]}(h) \\
[\![(\mathsf{i}, s)\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[\mathsf{i} \mapsto \mathcal{A}(s)]}(h) \\
[\![f(\varphi_1, \ldots, \varphi_m)]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= f([\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(h), \ldots, [\![\varphi_m]\!]^{\mathcal{G}}_{\mathcal{A}}(h)) \\
[\![\mathbf{X}\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_{|h|+1}) \\
[\![\varphi_1 \mathbf{U} \varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= \sup_{i \geq 0} \min([\![\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_{|h|+i}), \min_{0 \leq j < i} [\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_{|h|+j}))
\end{aligned}
$$

[1] Notice that, differently from a wCGSii, a wCGSdoes not contain observation relations.

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

As in Chapter 5, we use classic abbreviations. We also use $\mathbf{A}\varphi$ as a shorthand for a universal quantification on strategies and bindings for all agents, followed by φ; this simulates the universal path quantifier of CTL^*.

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

We investigate the following satisfiability problem for $\mathsf{SL}[\mathcal{F}]$.

Definition 7.4. The *satisfiability problem* for $\mathsf{SL}[\mathcal{F}]$ takes a sentence $\varphi \in \mathsf{SL}[\mathcal{F}]$ and a threshold $\varepsilon > -1$, and decides whether there exists a wCGS $\mathcal{G}$ such that $[\![\varphi]\!]^{\mathcal{G}} \geq \varepsilon$.

The satisfiability problem for SL was proved undecidable in (Mogavero et al., 2017), but the proof there considers models with infinitely many actions. However it is also known to be undecidable when considering finite models, *i.e.*, models with both finitely many states and finitely many actions, as we do. Indeed, it has been shown in (Troquard and Walther, 2012; Laroussinie and Markey, 2015) that satisfiability of ATL with strategy context ($\mathsf{ATL}_{\mathrm{sc}}$) is undecidable for finite models. Since $\mathsf{ATL}_{\mathrm{sc}}$ can be expressed in SL (Laroussinie and Markey, 2015) and SL in $\mathsf{SL}[\mathcal{F}]$, by taking $\mathcal{F} = \{\top, \vee, \neg\}$ (Bouyer et al., 2019), we obtain the following result.

Proposition 7.1. *The satisfiability problem for* $\mathsf{SL}[\mathcal{F}]$ *is undecidable as soon as* $\mathcal{F}$ *contains* $\top$, $\vee$ *and* $\neg$.

However satisfiability of SL is known to be decidable when restricted to turn-based systems or systems with a bounded number of actions (Laroussinie and Markey, 2015). We show that in these cases satisfiability is decidable for $\mathsf{SL}[\mathcal{F}]$ as well. To do so we first recall Booleanly-quantified CTL^* ($\mathsf{BQCTL}^*[\mathcal{F}]$) and solve its satisfiability problem, and we then show that for bounded actions or turn based systems satisfiability of $\mathsf{SL}[\mathcal{F}]$ reduces to that of $\mathsf{BQCTL}^*[\mathcal{F}]$.

7.2.1 Booleanly-Quantified $\mathsf{CTL}^*[\mathcal{F}]$

The logic $\mathsf{BQCTL}^*[\mathcal{F}]$ (Bouyer et al., 2019), a quantitative extension of QCTL^* (Laroussinie and Markey, 2014), itself an extension of CTL^* with quantifiers on atomic proposition. In $\mathsf{BQCTL}^*[\mathcal{F}]$ the semantics is quantitative, but the quantifiers on propositions consider only Boolean values. To be consistent with $\mathsf{SL}[\mathcal{F}]$ we consider [-1,1] as range of values instead of [0,1] as in (Bouyer et al., 2019). This changes nothing to the results presented there.

The syntax of $\mathsf{BQCTL}^*[\mathcal{F}]$ is defined by:

$$\varphi ::= p \mid \exists p.\, \varphi \mid \mathbf{E}\psi \mid f(\varphi, \ldots, \varphi)$$
$$\psi ::= \varphi \mid \mathbf{X}\psi \mid \psi \mathbf{U} \psi \mid f(\psi, \ldots, \psi)$$

where p ranges over AP and f over $\mathcal{F}$.

$\mathbf{E}\psi$ is the quantitative counterpart to the path quantifier of CTL^*, and it maximizes the value of ψ over all branches. Formulas of type φ are called *state formulas*, those of type ψ are called *path formulas*, and $\mathsf{BQCTL}^*[\mathcal{F}]$ consists of all the state formulas defined by the grammar. We again use $\top$, $\vee$, and $\neg$ to denote functions 1, max and $-x$, as well as classic abbreviations already introduced for $\mathsf{SL}[\mathcal{F}]$.

Definition 7.5. A *weighted Kripke structure* (wKS) is a tuple $\mathcal{S} = (S, s_\iota, R, \ell)$ where S is a set of states, $s_\iota \in S$ is an initial state, $R \subseteq S \times S$ is a left-total[2] transition relation, and $\ell\colon S \to [-1,1]^{\mathrm{AP}}$ is a weight function.

A *path* in $\mathcal{S}$ is an infinite word $\lambda = s_0 s_1 \dots$ over S such that $s_0 = s_\iota$ and $(s_i, s_{i+1}) \in R$ for all i. Finite prefixes of paths are *histories*, and we let $h_{\mathcal{S}}$ be the set of all histories in $\mathcal{S}$. We also let $\mathrm{Val}_{\mathcal{S}} = \{\ell(s)(p) \mid s \in S \text{ and } p \in \mathrm{AP}\}$ be the finite set of values appearing in $\mathcal{S}$.

Given finite nonempty sets X of directions and $\mathcal{V} \subseteq [-1,1]$ of possible values, a $\mathcal{V}^{\mathrm{AP}}$-labeled X-tree, (or $(\mathcal{V}^{\mathrm{AP}}, X)$-tree for short, or $\mathcal{V}^{\mathrm{AP}}$-tree when directions are understood), is a pair $t = (\tau, \ell)$ where $\tau \subseteq X^+$ is closed under non-empty prefixes, all *nodes* $u \in \tau$ start with the same direction r, called the *root*, and have at least one *child* $u \cdot d \in \tau$, and $\ell : \tau \to \mathcal{V}^{\mathrm{AP}}$ is a *weight function*. We let $\mathrm{Val}_t \subseteq \mathcal{V}$ be the image of ℓ on τ. A *branch* $\lambda = u_0 u_1 \dots$ is an infinite sequence of nodes such that for all $i \geq 0$, u_{i+1} is a child of u_i. For $i \geq 0$, $\lambda_{\geq i}$ denotes the suffix of λ that starts at node u_i, and we let $\mathrm{Br}(u)$ be the set of branches that start in node u.

A *binary tree* is a X-tree where $|X| = 2$. A tree is *regular* if it is the unfolding of some finite Kripke structure.

Let $p \in \mathrm{AP}$. A *p-labeling* for a $\mathcal{V}$-tree $t = (\tau, \ell)$ is a mapping $\ell_p : \tau \to \{-1, 1\}$. The composition of t with ℓ_p is the $(\mathcal{V} \cup \{-1,1\})^{\mathrm{AP}}$-tree defined as $t \otimes \ell_p =_{\mathrm{def}} (\tau, \ell')$, where $\ell'(u)(p) = \ell_p(u)$ and $\ell'(u)(q) = \ell(u)(q)$ for $q \neq p$.

Finally, the *tree unfolding of a weighted Kripke structure* $\mathcal{S}$ with state set S is the $\mathrm{Val}_{\mathcal{S}}^{\mathrm{AP}}$-labeled S-tree $t_{\mathcal{S}} = (h_{\mathcal{S}}, \ell')$, where $\ell'(u) = \ell(\mathrm{last}(u))$ for every $u \in h_{\mathcal{S}}$.

Different semantics exist for QCTL^*, which differ on how proposition quantification is interpreted. Notably, in the *structure semantics*, labelings for quantified propositions are defined directly on the states of the structure, while in the *tree semantics*, labelings are defined on the tree unfolding (see (Laroussinie and Markey, 2014) for more on the different semantics for QCTL^*). In this paper we focus on the tree semantics, which allows capturing perfect-recall strategies.

Definition 7.6 (Semantics)**.** Consider a finite set $\mathcal{V} \subseteq [-1,1]$ of possible values. The satisfaction value $[\![\varphi]\!]^t(u)$ of a $\mathsf{BQCTL}^*[\mathcal{F}]$ state formula φ in a node u of a $\mathcal{V}^{\mathrm{AP}}$-tree $t = (\tau, \ell)$, and the satisfaction value $[\![\psi]\!]^t(\lambda)$ of a path formula ψ along

[2] *i.e.*, for all $s \in S$, there exists s' such that $(s, s') \in R$.

some branch λ of t, are defined inductively as follows:

$$\begin{aligned}
[\![p]\!]^t(u) &= \ell(u)(p) \\
[\![\exists p.\, \varphi]\!]^t(u) &= \sup_{\ell_p : \tau \to \{-1,1\}} [\![\varphi]\!]^{t \otimes \ell_p}(u) \\
[\![\mathbf{E}\psi]\!]^t(u) &= \sup_{\lambda \in \mathrm{Br}(u)} [\![\psi]\!]^t(\lambda) \\
[\![f(\varphi_1, \ldots, \varphi_n)]\!]^t(u) &= f([\![\varphi_1]\!]^t(u), \ldots, [\![\varphi_n]\!]^t(u)) \\
[\![\varphi]\!]^t(\lambda) &= [\![\varphi]\!]^t(\lambda_0) \\
[\![\mathbf{X}\psi]\!]^t(\lambda) &= [\![\psi]\!]^t(\lambda_{\geq 1}) \\
[\![\psi_1 \mathbf{U} \psi_2]\!]^t(\lambda) &= \sup_{i \geq 0} \min([\![\psi_2]\!]^t(\lambda_{\geq i}), \min_{0 \leq j < i} [\![\psi_1]\!]^t(\lambda_{\geq j})) \\
[\![f(\psi_1, \ldots, \psi_n)]\!]^t(\lambda) &= f([\![\psi_1]\!]^t(\lambda), \ldots, [\![\psi_n]\!]^t(\lambda))
\end{aligned}$$

For a tree t with root r we write $[\![\varphi]\!]^t$ for $[\![\varphi]\!]^t(r)$, for a weighted Kripke structure $\mathcal{S}$ we write $[\![\varphi]\!]^{\mathcal{S}}$ for $[\![\varphi]\!]^{t_{\mathcal{S}}}$.

7.2.2 Deciding $\mathsf{BQCTL}^*[\mathcal{F}]$ Satisfiability

As for $\mathsf{SL}[\mathcal{F}]$, we define the quantitative satisfiability problem for $\mathsf{BQCTL}^*[\mathcal{F}]$ as follows.

Definition 7.7. The *satisfiability problem* for $\mathsf{BQCTL}^*[\mathcal{F}]$ takes a formula $\varphi \in \mathsf{BQCTL}^*[\mathcal{F}]$ and a threshold $\varepsilon > -1$, and decides whether there exists a wKS $\mathcal{S}$ s.t. $[\![\varphi]\!]^{t_{\mathcal{S}}} \geq \varepsilon$.

Satisfiability for QCTL^* with structure semantics is undecidable (French, 2003), but decidable for the tree semantics (Laroussinie and Markey, 2014) which we consider in this paper. Relying on the automata construction developed in (Bouyer et al., 2019) to model check $\mathsf{BQCTL}^*[\mathcal{F}]$, we show that satisfiability is also decidable for $\mathsf{BQCTL}^*[\mathcal{F}]$.

Theorem 7.1. *Satisfiability of* $\mathsf{BQCTL}^*[\mathcal{F}]$ *is decidable.*

In the rest of the section we sketch the proof of this theorem, and refer to Appendix B for the full proof.

The lower bounds are inherited from the satisfiability problem for QCTL^* (Laroussinie and Markey, 2014). The reduction is direct with threshold $\varepsilon = 1$.

For the upper bounds, the first step is to show that one can restrict attention to structures with binary branching, *i.e.*, where each state has at most two successor states.

Given a finite wKS $\mathcal{S}$, by adding dummy nodes labeled with proposition p_{int} one can build a wKS $\widetilde{\mathcal{S}}$ with binary branching that simulates $\mathcal{S}$; given a $\mathsf{BQCTL}^*[\mathcal{F}]$ formula φ, one can then define a formula $\widetilde{\varphi}$ that ignores these dummy nodes and has same satisfaction value on $\widetilde{\mathcal{S}}$ as φ on $\mathcal{S}$.

Lemma 7.1. *For every* $\mathsf{BQCTL}^*[\mathcal{F}]$ *formula* φ *and every finite wKS* $\mathcal{S}$, $[\![\varphi]\!]^{\mathcal{S}} = [\![\widetilde{\varphi}]\!]^{\widetilde{\mathcal{S}}}$.

Let Φ be a $\mathsf{BQCTL}^*[\mathcal{F}]$ formula and $\varepsilon > -1$ a threshold. Let also $\widetilde{\Phi}$ be the corresponding formula on structures with binary branching, and $\varphi_2 = \neg p_{\text{int}} \wedge \mathbf{AGF}\neg p_{\text{int}}$. Consider also function Bool such that $\text{Bool}(x) = 1$ if $x \in \{-1, 1\}$, and -1 otherwise, which we use to check that propositions take only Boolean values -1 or 1. We can prove the following:

Lemma 7.2. *There exists a (finite) wKS* $\mathcal{S}$ *such that* $[\![\Phi]\!]^{\mathcal{S}} \geq \varepsilon$ *if and only if there exists a regular binary tree* t *such that* $[\![\mathbf{AG}\text{Bool}(p_{\text{int}}) \wedge \varphi_2 \wedge \widetilde{\Phi}]\!]^t \geq \varepsilon$.

It is shown in (Bouyer et al., 2019) that we can build an automaton $\mathcal{A}$ that accepts binary trees t on which the satisfaction value of $\mathbf{AG}\text{Bool}(p_{\text{int}}) \wedge \varphi_2 \wedge \widetilde{\Phi}$ belongs to $[\varepsilon, 1]$ (we refer to (Pin, 2021, Chapter 8) for standard definitions and results on tree automata). The number of states of the automaton is $(k+1)$-exponential in the nesting depth of the formula, and its index k-exponential. From Lemma 7.2 and the fact that every regular tree language contains a regular tree it follows that $\mathcal{A}$ is nonempty if and only if there exists a finite wKS $\mathcal{S}$ such that $[\![\Phi]\!]^t \geq \varepsilon$. The emptiness of $\mathcal{A}$ can be tested in time polynomial in the number of states and exponential in the index. The overall complexity is thus $(k+1)$-exponential in time.

7.2.3 Decidable Cases for $\mathsf{SL}[\mathcal{F}]$ Satisfiability

In the qualitative setting, satisfiability of ATL with strategy context and SL are known to be decidable in two cases: when the number of possible actions is bounded, and when agents play in turns. With respect to bounded actions, we call $\mathcal{B}$-wCGS a wCGS whose set of actions is $\mathcal{B}$. Concerning turn-based systems, intuitively a wCGS is turn-based when in every position there is a unique agent who can determine the next position by the choice of its action. Formally, a wCGS $\mathcal{G} = (\mathcal{B}, V, v_\iota, \delta, \ell)$ is *turn-based* if for every position $v \in V$ there exists a unique agent i such that for every successor $v' \in \{\delta(v, \boldsymbol{a}) \mid \boldsymbol{a} \in \mathcal{B}^{\mathrm{N}}\}$ there is an action a such that $\delta(v, \boldsymbol{a}) = v'$ for all joint actions $\boldsymbol{a}$ where $\boldsymbol{a}_{\mathsf{i}} = a$. We call *owner* of a position the agent that can choose the successor.

We show that, as for SL, the satisfiability problem for $\mathsf{SL}[\mathcal{F}]$ is decidable when the number of actions is bounded (*a priori*) or systems are turn-based, if in addition we restrict to models in which atomic propositions take values in a given finite set of possible values. Remind we consider a finite model, which has a finite (and thus bounded) number of actions. However the set of all finite models contains models with arbitrarily large sets of actions. The bound on the number of actions is put on the set of models considered for the satisfiability problem. Given a set $\mathcal{V} \subseteq [-1, 1]$ of possible values with $\{-1, 1\} \subseteq \mathcal{V}$, we call $\mathcal{V}$-*weighted* wCGS a wCGS whose weight function takes values in $\mathcal{V}$.

Definition 7.8. Given a finite set $\mathcal{V} \subset [-1,1]$ such that $\{-1,1\} \subseteq \mathcal{V}$, the $\mathcal{V}$-*satisfiability problem* for $\mathsf{SL}[\mathcal{F}]$ is the restriction of the satisfiability problem to $\mathcal{V}$-weighted wCGS.

We establish the following results:

Theorem 7.2. *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 $\mathcal{B}$-wCGS is decidable.*

Proof. We use a slight modification of the proof in Laroussinie and Markey (2015) which shows how, when the number of actions is bounded, one can reduce the satisfiability problem for SL to that for QCTL^*.

Let $\mathcal{B} = \{a_1, \dots, a_n\}$ and $\mathrm{N} = \{\mathsf{i}_1, \dots, \mathsf{i}_m\}$. First, consider the formula

$$\Phi_{\mathrm{Bool}} = \mathbf{AG} \bigwedge_{\mathsf{i} \in \mathrm{N}, a \in \mathcal{B}} \mathrm{Bool}(\mathrm{mov}_{\mathsf{i}}^{a})$$

This formula has value 1 in a tree if propositions $\mathrm{mov}_{\mathsf{i}}^{a}$ only take Boolean values in all nodes of the tree, otherwise it has value -1. Define also

$$\Phi_{\mathrm{edge}} = \mathbf{AG}\left[\left(\bigwedge_{\boldsymbol{a} \in \mathcal{B}^{\mathrm{N}}} \mathbf{E}_1 \mathbf{X} \mathrm{mov}_{\boldsymbol{a}} \right) \wedge \mathbf{AX}\left(\bigvee_{\boldsymbol{a} \in \mathcal{B}^{\mathrm{N}}} \mathrm{mov}_{\boldsymbol{a}} \right) \right]$$

where $\mathrm{mov}_{\boldsymbol{a}}$ stands for $\bigwedge_{\mathsf{i} \in \mathrm{N}} (\mathrm{mov}_{\mathsf{i}}^{a_{\mathsf{i}}} \wedge \bigwedge_{a' \neq a_{\mathsf{i}}} \neg \mathrm{mov}_{\mathsf{i}}^{a'})$, and

$$\mathbf{E}_1 \mathbf{X} \varphi = \mathbf{EX} \varphi \wedge \forall p. (\mathbf{EX}(\varphi \wedge p) \rightarrow \mathbf{AX}(\varphi \rightarrow p))$$

expresses the existence of a unique successor that satisfies φ (where φ is a formula that only takes Boolean values). When all propositions $\mathrm{mov}_{\mathsf{i}}^{a}$ have Boolean value, Φ_{edge} can only take value 1 or -1 in a tree, and this value is 1 if and only if the tree is the unfolding of some wCGS where in every position, proposition $\mathrm{mov}_{\mathsf{i}}^{a}$ has value 1 if agent i played action a in the last move, and -1 otherwise.

Consider now the following translation from $\mathsf{SL}[\mathcal{F}]$ to $\mathsf{BQCTL}^*[\mathcal{F} \cup \{\mathrm{Bool}\}]$. For every $\mathsf{SL}[\mathcal{F}]$ formula φ and partial function $V : \mathrm{N} \to \mathrm{Var}$, we inductively define the $\mathsf{BQCTL}^*[\mathcal{F}]$ formula $\widehat{\varphi}^V$. The translation is identical to the one in Laroussinie and Markey (2015), but for completeness we give the cases for strategy quantification and binding, as well as temporal operators.

$$\begin{aligned}
\widehat{\exists s. \varphi}^V &= \exists p_s^{a_1} \dots \exists p_s^{a_n} . \mathbf{AG} \left(\bigvee_{a \in \mathcal{B}} p_s^a \wedge \bigwedge_{a' \neq a} \neg p_s^{a'} \right) \wedge \widehat{\varphi}^V \\
\widehat{(\mathsf{i}, s)\varphi}^V &= \widehat{\varphi}^{V[\mathsf{i} \mapsto s]} \\
\widehat{\mathbf{X}\varphi}^V &= \mathbf{A}\left[\psi_{\mathrm{out}}^V \rightarrow (\mathbf{X} \widehat{\varphi}^V) \right] \\
\widehat{\varphi \mathbf{U} \psi}^V &= \mathbf{A}\left[\psi_{\mathrm{out}}^V \rightarrow (\widehat{\varphi}^V \mathbf{U} \widehat{\psi}^V) \right]
\end{aligned}$$

where $\psi^V_{\text{out}} = \mathbf{G} \bigwedge_{\mathsf{i}\in \mathrm{N}} \bigwedge_{a\in\mathcal{B}} (p^a_{V(\mathsf{i})} \to \mathbf{X}\mathrm{mov}^a_{\mathsf{i}})$.

Finally we let $\widehat{\Phi} = \Phi_{\text{Bool}} \wedge \Phi_{\text{edge}} \wedge \widehat{\Phi}^{\emptyset}$. We have that for every finite wCGS$\mathcal{G}$ there exists a finite wKS $\mathcal{S}$ such that $[\![\Phi]\!]^{\mathcal{G}} = [\![\widehat{\Phi}]\!]^{\mathcal{S}}$, where $\mathcal{S}$ is obtained by partially unfolding $\mathcal{G}$ to place propositions $\mathrm{mov}^a_{\mathsf{i}}$ appropriately. Conversely, for every finite $\mathcal{S}$ such that $[\![\widehat{\Phi}]\!]^{\mathcal{S}} > -1$ we have that $[\![\Phi_{\text{Bool}} \wedge \Phi_{\text{edge}}]\!]^{\mathcal{S}} = 1$ and thus $\mathcal{S}$ encodes a finite wCGS$\mathcal{G}$ such that $[\![\Phi]\!]^{\mathcal{G}} = [\![\widehat{\Phi}]\!]^{\mathcal{S}}$. □

Theorem 7.3. *Let $\mathcal{V}$ be a finite set of values. Then $\mathcal{V}$-satisfiability of $\mathsf{SL}[\mathcal{F}]$ over turn-based wCGS is decidable.*

Proof. Again, we adapt the proof from Laroussinie and Markey (2015). We assume that in a turn-based wCGS, each position is labelled with a proposition $\mathrm{turn}_{\mathsf{i}}$ where i is the owner of the position. We use formula

$$\varphi_{\text{tb}} = \mathbf{AG}\left(\bigwedge_{\mathsf{i}\in\mathrm{N}} \mathrm{Bool}(\mathrm{turn}_{\mathsf{i}}) \wedge \bigvee_{\mathsf{i}\in\mathrm{N}} (\mathrm{turn}_{\mathsf{i}} \wedge \bigwedge_{\mathsf{i}'\neq\mathsf{i}} \neg\mathrm{turn}_{\mathsf{i}'})\right)$$

that has Boolean value, and has value 1 if and only if propositions $\mathrm{turn}_{\mathsf{i}}$ only take Boolean values (-1 or 1), and exactly one of them has value 1 in each node.

We now define the following translation from $\mathsf{SL}[\mathcal{F}]$ to $\mathsf{BQCTL}^*[\mathcal{F}]$, where $V : \mathrm{N} \to \mathrm{Var}$ is a partial function.

$$\begin{aligned}
\widehat{\exists s.\, \varphi}^V &= \exists \mathrm{mov}_s.\mathbf{AG}(\mathbf{E}_1\mathbf{X}\mathrm{mov}_s \wedge \widehat{\varphi}^V)\\
\widehat{(\mathsf{i}, s)\varphi}^V &= \widehat{\varphi}^{V[\mathsf{i}\mapsto s]}\\
\widehat{\mathbf{X}\varphi}^V &= \mathbf{A}\left[\psi^V_{\text{out}} \to (\mathbf{X}\widehat{\varphi}^V)\right]\\
\widehat{\varphi\mathbf{U}\psi}^V &= \mathbf{A}\left[\psi^V_{\text{out}} \to (\widehat{\varphi}^V\mathbf{U}\widehat{\psi}^V)\right]
\end{aligned}$$

where $\psi^V_{\text{out}} = \mathbf{G}(\bigwedge_{\mathsf{i}\in\mathrm{N}}(\mathrm{turn}_{\mathsf{i}} \to \mathbf{X}\mathrm{mov}_{\mathsf{i}}))$. Finally, formula Φ has value greater than ε in some turn-based wCGSif and only if formula $\varphi_{\text{tb}} \wedge \widehat{\Phi}^{\emptyset}$ has value greater than ε in some tree. □

Concerning complexity, for Theorems 7.1, 7.2 and 7.3 the problems are $(k+1)$-EXPTIME-complete where k is the maximal number of nested quantifiers on propositions (for $\mathsf{BQCTL}^*[\mathcal{F}]$) or strategies (for $\mathsf{SL}[\mathcal{F}]$). Blocks of successive quantifiers can be counted as one if they are all existential or all universal, so that an existential quantification on a strategy profile for all agents just adds one exponential for instance.

7.2.4 Automated Synthesis of Optimal Mechanism

We describe how we can use our 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 observe that the algorithms developed in the previous section for the satisfiability problem of $\mathsf{SL}[\mathcal{F}]$ in the case of bounded actions or turn-based systems can be tweaked to actually return a satisfying wCGS when one exists. Indeed classic algorithms to solve emptiness of parity tree automata can produce a witness regular tree accepted by the automaton (see for instance Pnueli and Rosner (1989)), and from such a regular tree in our setting we can infer a witness wCGS. In particular, once we have a regular tree represented as a finite transition system with states labelled with atomic propositions, one easily obtains the desired wCGS by seeing states as positions and keeping the same labelling for "normal" atomic propositions. Concerning transitions, in the case of bounded actions, the labelling for "special" propositions mov_a^c induces a transition function for action profiles. In the turn-based case, turns are described by the labelling for the "special" propositions turn_a; one can then use one action for each possible position, and a dummy action for inactive agents, to represent possible choices of agents and define the transition function. Finally one can remove the dummy internal positions (labelled with p_{int}) introduced in the reduction to binary branching. The synthesis procedure produces models with abstract actions, but their "real world" meaning can be recovered thanks to the propositions labelling (assuming the specification was precise enough).

Second, it is proved in (Bouyer et al., 2019) 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 of this set.

Lemma 7.3 (Bouyer et al. (2019))**.** *Let $\mathcal{V} \subset [-1,1]$ be a finite set of values with $\{-1,1\} \subseteq \mathcal{V}$ and let φ be an $\mathsf{SL}[\mathcal{F}]$ sentence. The set $\mathrm{Val}_{\varphi,\mathcal{V}} = \{[\![\varphi]\!]^{\mathcal{G}} \mid \mathcal{G}$ is a $\mathcal{V}$-weighted wCGS$\}$ is finite, and one can compute a set $\widetilde{\mathrm{Val}}_{\varphi,\mathcal{V}}$ of size at most $|\mathcal{V}|^{|\varphi|}$ such that $\mathrm{Val}_{\varphi,\mathcal{V}} \subseteq \widetilde{\mathrm{Val}}_{\varphi,\mathcal{V}}$.*

Algorithm 3 *synthesis*$(\Phi, \mathcal{V})$

Input: a $\mathsf{SL}[\mathcal{F}]$ specification 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 $\widetilde{\mathrm{Val}}_{\Phi,\mathcal{V}}$
2: Let $\nu_1, \ldots, \nu_n$ be a decreasing enumeration of $\widetilde{\mathrm{Val}}_{\Phi,\mathcal{V}}$
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 $\mathcal{G}$

Consider now Algorithm 3. This algorithm 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. In particular, it works in the case of bounded actions and the case of turn-based systems. We now show how this can be used to solve automated mechanism design.

7.3 Synthesis for Mechanism Design

This section aims to motivate the use of synthesis of $\mathsf{SL}[\mathcal{F}]$ 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. Since many mechanisms describe monetary transfers, we assume that each alternative is a tuple (x, p) where $x \in \mathcal{X}$ is a *choice* from a finite set of choices $\mathcal{X} \subset [-1, 1]$, and $\mathrm{p}_\mathsf{i} \in [-1, 1]$ is the payment for agent i. For each agent $\mathsf{i} \in \mathrm{N}$, let also $\Theta_\mathsf{i} \subset [-1, 1]$ be a finite set of possible *types* for i. We let $\boldsymbol{\Theta} = \prod_{\mathsf{i} \in \mathrm{N}} \Theta_\mathsf{i}$, and we note $\boldsymbol{\theta} = (\theta_\mathsf{i})_{\mathsf{i} \in \mathrm{N}} \in \boldsymbol{\Theta}$ 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* $\mathsf{v}_\mathsf{i} : \mathcal{X} \times \Theta_\mathsf{i} \to [-1, 1]$.

In this section we assume that $\mathcal{F}$ contains the constant θ_i and the function v_i for each agent i and type $\theta_\mathsf{i} \in \Theta_\mathsf{i}$. We also assume that $\mathcal{F}$ contains the difference function $-$, the n-ary sum function $\sum$, the equality function $=$ and the comparison functions $\leq$, $>$, $<$ and $\neq$. For readability we use the infix notation $x - y$. These functions are defined with the standard meaning. Finally, we assume that types, payments and valuations are normalized so that all values remain in $[-1, 1]$.

A mechanism consists of a description of the agents' possible strategies, and a description of the alternatives that result from them. Similar to Chapter 5[3], we can represent mechanisms as wCGS and verify their equilibrium outcome in relation to a number of economic properties.

Definition 7.9. Let $\mathrm{AP} \supseteq \{\mathrm{choice}, \mathrm{pay}_\mathsf{i}, \mathrm{terminal} : \mathsf{i} \in \mathrm{N}\}$, where choice and pay_i denote respectively the choice elected by the mechanism, and the payment of agent i. The proposition terminal specifies whether a position is terminal. A *mechanism* $\mathcal{G}$ is a wCGS over the atomic propositions AP that satisfies the following:

(i) every play eventually reaches a *terminal position*, *i.e.*, a sink;

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

7.3.1 Characterizing Properties with $\mathsf{SL}[\mathcal{F}]$

$\mathsf{SL}[\mathcal{F}]$ can express a variety of important notions in mechanism design, such as strategyproofness, individual rationality, efficiency, budget-balance, Pareto optimality, and different kinds of game-theoretic equilibria (see Chapter 5). We recall the formulas for some of these notions. Let $\boldsymbol{\theta} = (\theta_\mathsf{i})_{\mathsf{i} \in \mathrm{N}}$ be a type profile in $\boldsymbol{\Theta}$.

First define $\mathrm{util}_\mathsf{i}(\theta_\mathsf{i}) =_{\mathrm{def}} \mathsf{v}_\mathsf{i}(\mathrm{choice}, \theta_\mathsf{i}) - \mathrm{pay}_\mathsf{i}$. This is an $\mathsf{SL}[\mathcal{F}]$ formula, whose value in a terminal position is equal to the agent's utility, which she tries to maximize.

We recall that the $\mathsf{SL}[\mathcal{F}]$ formula for encoding individual rationality is

$$\mathrm{IR}(\boldsymbol{\theta}) =_{\mathrm{def}} \bigwedge_{\mathsf{i} \in \mathrm{N}} 0 \leq \mathrm{util}_\mathsf{i}(\theta_\mathsf{i})$$

[3]The definitions in this chapter are simplified since we do not encode imperfect information in the wCGS.

Similarly, efficiency is expressed as follows:

$$\mathrm{EF}(\boldsymbol{\theta}) =_{\mathrm{def}} \sum_{\mathsf{i}\in\mathrm{N}} \mathsf{v}_{\mathsf{i}}(\mathrm{choice}, \theta_{\mathsf{i}}) = \mathrm{maxv}_{\boldsymbol{\theta}}$$

where $\mathrm{maxv}_{\boldsymbol{\theta}} = \max_{x\in\mathcal{X}} \sum_{\mathsf{i}\in\mathrm{N}} \mathsf{v}_{\mathsf{i}}(x, \theta_{\mathsf{i}})$ is a constant in $\mathcal{F}$. In a terminal position, it means that the social welfare is maximal.

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

$$\mathrm{NE}(\boldsymbol{s}, \boldsymbol{\theta}) =_{\mathrm{def}} \bigwedge_{\mathsf{i}\in\mathrm{N}} \forall t.\, [(\mathrm{N}_{-\mathsf{i}}, \boldsymbol{s}_{-\mathsf{i}})(\mathsf{i}, t)\mathbf{F}(\mathrm{terminal} \wedge \mathrm{util}_{\mathsf{i}}(\theta_{\mathsf{i}})) \\ \leq(\mathrm{N}, \boldsymbol{s})\mathbf{F}(\mathrm{terminal} \wedge \mathrm{util}_{\mathsf{i}}(\theta_{\mathsf{i}}))]$$

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

The following formula maximizes the value of φ in the terminal positions of all Nash equilibria:

$$\text{Max-Nash}(\varphi, \boldsymbol{\theta}) =_{\mathrm{def}} \exists \boldsymbol{s}.\, \mathrm{NE}(\boldsymbol{s}, \boldsymbol{\theta}) \wedge \mathbf{F}(\mathrm{terminal} \wedge \varphi)$$

We now use these formulas to illustrate our approach to mechanism synthesis. To avoid detailing tie-breaking rules, in the examples we assume agents have distinct types, that is $\theta_{\mathsf{i}} \neq \theta_{\mathsf{r}}$ for any $\mathsf{r} \neq \mathsf{i}$ and $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. Given an agent i, we let $\mathrm{wins}_{\mathsf{i}} \in (-1, 1]$ be a constant value denoting the choice in which i is the winner, with $\mathrm{wins}_{\mathsf{i}} \neq \mathrm{wins}_{\mathsf{r}}$ for any $\mathsf{r} \neq \mathsf{i}$. We consider the choice set $\mathcal{X} = \{\mathrm{wins}_{\mathsf{i}} : \mathsf{i} \in \mathrm{N}\} \cup \{-1\}$, where -1 specifies the case where there is no winner at the end of the game. In the examples we let each valuation function v_{i} be defined as $\mathsf{v}_{\mathsf{i}}(\theta_{\mathsf{i}}, x) = \theta_{\mathsf{i}}$ if $x = \mathrm{wins}_{\mathsf{i}}$, and $\mathsf{v}_{\mathsf{i}}(\theta_{\mathsf{i}}, x) = 0$ otherwise. That is, the valuation of an agent depends only on her type and whether she won.

7.3.2 Action-bounded Mechanisms

Action-bounded mechanisms are of great interest since the amount of resources available is often limited. For instance, the actions in a market could consist in bids representing discrete monetary values bounded by the participants' budget. Besides that, there are games where the available budget is limited due to fairness constraints, *e.g.*, sports clubs negotiating the hiring of professional players. Furthermore, a number of mechanisms have small range of available actions. For instance, in the Dutch and Japanese auctions, players can only bid "accept" or "reject" in each turn. In this section, we illustrate the synthesis problem with such restriction by considering rules based on the Japanese auction.

Example 7.1. The Japanese auction is an ascending protocol in which the price is repeatedly raised by the auctioneer until only one bidder remains. The remaining bidder wins the item at the final price (Klemperer, 1999). 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

$\Phi = \{\text{price}, \text{sold}, \text{initial}, \text{choice}, \text{bid}_\mathsf{i}, \text{pay}_\mathsf{i}, \text{terminal} : \mathsf{i} \in \text{N}\}$, where price denotes the current price, initial denotes whether the position is the initial one, sold specifies whether the item was sold, and bid_i specifies whether i is an active bidder.

The following $\mathsf{SL}[\mathcal{F}]$-formulae are a partial description of a mechanism, inspired by the Japanese auction. Similar rules for encoding auctions through ADL can be seen in Chapter 3. Rule J1 says that proposition initial implies the position is not terminal and the price is zero. Additionally, the auction will eventually reach a position in which terminal will be true and that proposition initial can only be true in the initial position. Rule J2 says the good is sold whenever the proposition choice does not have the value -1. Rules J3-J4 specifies the price update. Rule J5 chooses a player as the winner if she is the only one bidding for the item. When no agent can be chose as the winner, the choice is -1 (Rule J6). Rules J7-J8 specify the payment for each agent, which is either equal to the price or zero, depending on whether the agent won the auction. Rule J9 specifies that for all type profiles there should exist a NE whose outcome is IR and EF.

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

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

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

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

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

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

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

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

J9. $\bigwedge_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \text{Max-Nash}(\text{IR}(\boldsymbol{\theta}) \wedge \text{EF}(\boldsymbol{\theta}), \boldsymbol{\theta})$

We denote by Σ_{jpn} the conjunction of Rules J1-J9. Algorithm 3 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.

Proposition 7.2. *There exists a wCGS-mechanism $\mathcal{G}_{\text{jpn}}$ such that $[\![\Sigma_{\text{jpn}}]\!]^{\mathcal{G}_{\text{jpn}}} = 1$.*

Proof. We construct a wCGS-mechanism $\mathcal{G}_{\text{jpn}}$ in which Σ_{jpn} has the satisfaction value equal to one, for any position and assignment.

Let $\mathcal{G}_{\text{jpn}} = (\mathcal{B}, V, \delta, \ell, v_\iota)$ over $\text{AP} = \{\text{price}, \text{sold}, \text{initial}, \text{wins}_\mathsf{i}, \text{bid}_\mathsf{i}, \text{pay}_\mathsf{i}, \text{terminal} : \mathsf{i} \in \text{N}\}$, where:

- V consists of positions of the form $\langle \text{pr}, \text{buyer}, \theta_1, \ldots, \theta_\mathsf{n} \rangle$ with $\text{p} \in \{0 + x \cdot \mathsf{inc} : 0 \leq x \leq \frac{1}{\mathsf{inc}}\}$ denoting the current price, winner $\in \text{N} \cup \{\mathit{none}\}$ denoting the current winner, $\text{b}_\mathsf{i} \in \{-1, 1\}$ specifying whether i is an active bidder, and $\text{s} \in \{-1, 1\}$ specifying whether the item was sold.

- In an initial position, the price starts at 0, the item is unsold and there is no winner or active bidder. That is, the initial positions is $v_\iota = \{\langle 0, none, (-1, \ldots, -1), -1\rangle\}$.
- If the item is unsold, the transition function increases the price p as long as it is under 1 and there is at least one active bidder. If there is only one bidder, she is assigned as the winner. When the item is sold or the price reaches 1, the atomic propositions remain unchanged after the transition. Formally, for each position $v = \langle \mathrm{pr}, \mathrm{buyer}, \theta_1, \ldots, \theta_{\mathsf{n}}\rangle$ and joint action $\boldsymbol{a} = (a_{\mathsf{i}})_{\mathsf{i}\in \mathrm{N}}$, transition $\delta(v, \boldsymbol{a})$ is defined as follows:
  - If $\mathrm{s} = -1$ and $\mathrm{p} + \mathsf{inc} \leq 1$, $\delta(v, \boldsymbol{a}) = \langle \mathrm{pr}', \mathrm{buyer}', \theta_1, \ldots, \theta_{\mathsf{n}}\rangle$ where:

$$\mathrm{p}' = \mathrm{p} + \mathsf{inc}$$

$$\mathrm{winner}' = \begin{cases} \mathsf{i} & \text{if } a_{\mathsf{i}} = \mathrm{acc} \text{ and } a_{\mathsf{r}} = \mathrm{dec} \\ & \text{for all } \mathsf{r} \neq \mathsf{i} \\ none & \text{otherwise} \end{cases}$$

$$\mathrm{b}'_{\mathsf{i}} = \begin{cases} 1 & \text{if } a_{\mathsf{i}} = \mathrm{acc} \\ -1 & \text{otherwise} \end{cases}$$

$$\mathrm{s}' = \begin{cases} 1 & \text{if } \mathrm{winner}' \neq none \\ -1 & \text{otherwise} \end{cases}$$

  - Otherwise, $\delta(v, \boldsymbol{a}) = v$.
- For each $v = \langle \mathrm{pr}, \mathrm{buyer}, \theta_1, \ldots, \theta_{\mathsf{n}}\rangle$ and each $\mathsf{i} \in \mathrm{N}$, the weight function is defined as follows:
  - $\ell(v, \mathrm{price}) = \mathrm{p}$,
  - $\ell(v, \mathrm{choice}) = \mathrm{wins}_{\mathrm{winner}}$ iff $\mathrm{winner} \neq none$ and $\ell(v, \mathrm{wins}_{\mathsf{i}}) = -1$ otherwise,
  - $\ell(v, \mathrm{bid}_{\mathsf{i}}) = \mathrm{b}_{\mathsf{i}}$,
  - $\ell(v, \mathrm{sold}) = \mathrm{s}$,
  - $\ell(v, \mathrm{initial}_{\mathsf{i}}) = 1$ iff $v = v_\iota$ and $\ell(v, \mathrm{initial}_{\mathsf{i}}) = -1$ otherwise,
  - $\ell(v, \mathrm{pay}_{\mathsf{i}}) = \mathrm{p}$ iff $\mathrm{wins}_{\mathsf{i}} = \mathrm{winner}$ and $\ell(v, \mathrm{pay}_{\mathsf{i}}) = 0$ otherwise, and
  - $\ell(v, \mathrm{terminal}_{\mathsf{i}}) = 1$ iff $\mathrm{s} = -1$ and $\mathrm{p} + \mathsf{inc} \leq 1$ and $\ell(v, \mathrm{terminal}_{\mathsf{i}}) = -1$ otherwise.

It is straightforward to see that Rules J1-J8 are true in any position $v \in V$ and assignment $\mathcal{A}$. For J9, let $\pi \in \mathrm{Hist}$ be a history of length i. We show that there exists a strategy profile $\boldsymbol{\sigma}$ such that $[\![\exists \boldsymbol{s}.\, \mathrm{NE}(\boldsymbol{s}, \boldsymbol{\theta}) \wedge \mathbf{F}(\mathrm{terminal} \wedge (\mathrm{IR}(\boldsymbol{\theta}) \wedge \mathrm{EF}(\boldsymbol{\theta})))]\!]^{\mathcal{G}_{\mathrm{jpn}}}(\pi) = 1$. For each agent i, let σ_{i} be a strategy defined as follows: $\sigma_{\mathsf{i}}(\mathrm{last}(\pi)) = \mathrm{acc}$ iff $\mathsf{v}_{\mathsf{i}}(\mathrm{wins}_{\mathsf{i}}, \theta_{\mathsf{i}}) \geq \ell(\mathrm{last}(\pi), \mathrm{price}) + \mathsf{inc}$; and $\sigma_{\mathsf{i}}(v) = \mathrm{dec}$ otherwise.

For seeing this strategy is a Nash equilibrium, let $\mathcal{A}$ be an assignment such that $\mathcal{A}(\mathsf{i}) = \sigma_\mathsf{i}$ and $\mathcal{A}'$ be an assignment equal to $\mathcal{A}$, except by the strategy associated to agent i (that is, $\mathcal{A}'(\mathsf{i}) \neq \mathcal{A}(\mathsf{i})$).

The definition of $\mathcal{G}_{\text{jpn}}$ ensures the agents utility can be different from zero only in terminal positions. Furthermore, as this value depends only on the immediately preceding position and action, we consider the agent's utility after each transition. Let $\boldsymbol{\theta} = (\theta_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$ be a type profile in $\boldsymbol{\Theta}$. If $\mathcal{A}'(\mathsf{i})(\text{last}(\pi)) = \text{acc}$ when $\mathsf{v}_\mathsf{i}(\text{wins}_\mathsf{i}, \theta_\mathsf{i}) < \ell(\text{last}(\pi), \text{price}) + \mathsf{inc}$, then $[\![\mathbf{X}\text{util}_\mathsf{i}(\theta_\mathsf{i})]\!]^{\mathcal{G}_{\text{jpn}}}_{\mathcal{A}'}(\pi) \leq 0 \leq [\![\mathbf{X}\text{util}_\mathsf{i}(\theta_\mathsf{i})]\!]^{\mathcal{G}_{\text{jpn}}}_{\mathcal{A}}(\pi)$, since by doing the action acc the agent would be risking to get the item at a higher price than her valuation. Assume $\mathcal{A}'(\mathsf{i})(\pi) = \text{acc}$ and $\mathsf{v}_\mathsf{i}(\text{wins}_\mathsf{i}, \theta_\mathsf{i}) < \ell(\text{last}(\pi), \text{price}) + \mathsf{inc}$. By the definition of σ_i, $\mathcal{A}(\mathsf{i})(\text{last}(\pi)) = \text{dec}$ then $[\![\mathbf{X}\text{util}_\mathsf{i}(\theta_\mathsf{i})]\!]^{\mathcal{G}_{\text{jpn}}}_{\mathcal{A}'}(\pi) \leq 0$ and $[\![\mathbf{X}\text{util}_\mathsf{i}(\theta_\mathsf{i})]\!]^{\mathcal{G}_{\text{jpn}}}_{\mathcal{A}}(\pi) = 0$, since by doing the action acc the agent would be risking to get the item at a higher price than her valuation. Now consider the case in which $\mathcal{A}'(\mathsf{i})(\text{last}(\pi)) = \text{dec}$ and $\mathsf{v}_\mathsf{i}(\text{wins}_\mathsf{i}, \theta_\mathsf{i}) \geq \ell(\text{last}(\pi), \text{price}) + \mathsf{inc}$. By definition, $\mathcal{A}(\mathsf{i})(\text{last}(\pi)) = \text{acc}$. Thus, $[\![\mathbf{X}\text{util}_\mathsf{i}(\theta_\mathsf{i})]\!]^{\mathcal{G}_{\text{jpn}}}_{\mathcal{A}'}(\pi) = 0$, while $[\![\mathbf{X}\text{util}_\mathsf{i}(\theta_\mathsf{i})]\!]^{\mathcal{G}_{\text{jpn}}}_{\mathcal{A}}(\pi) \geq 0$, since by doing the action dec the agent would not win the item, whereas doing the action acc could lead to winning at a price lower than her valuation. The other cases are proven similarly. Since no agent can improve her utility through a unilateral change of strategy, the strategy profile $\boldsymbol{\sigma} = (\sigma_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$ is a Nash equilibrium. Furthermore, no agent i following the strategy σ_i can have a negative utility, since she chooses the action dec whenever the price would became higher than her valuation. From Rule J1, we have $[\![\mathbf{F}\text{terminal}]\!]^{\mathcal{G}_{\text{jpn}}}(\pi)$. Finally, since the agents have distinct valuations, following $\boldsymbol{\sigma}$ would lead to a position in which the item is assigned to the agent who valuates it the most. If no agent had a valuation for winning the item grater than 0, no winner would be assigned and the social welfare would be 0. Thus, $[\![\exists \boldsymbol{s}.\, \mathrm{NE}(\boldsymbol{s}, \boldsymbol{\theta}) \wedge \mathbf{F}(\text{terminal} \wedge (\mathrm{IR}(\boldsymbol{\theta}) \wedge \mathrm{EF}(\boldsymbol{\theta})))]\!]^{\mathcal{G}_{\text{jpn}}}(\pi) = 1$. □

Such an optimal mechanism is produced by Algorithm 3.

7.3.3 Turn-based Mechanisms

The turn-based restriction captures asynchronous bidding behaviour, which is considered, for instance, for investigating the convergence of bidding strategies for sequential keyword auctions (Cary et al., 2007). Additionally, in a number of mechanisms, such as picking sequences (Bouveret and Lang, 2014) and circle auctions (Schreiber and Romero, 2021), agents play in turns. Hereby, we exemplify the synthesis of a turned-based mechanism. In this example we also show how Algorithm 3 can be used to maximize the social welfare.

Example 7.2. The English auction is an ascending auction in which the participants are allowed to outbid the last bidder by proposing a highest price. The auction ends when no agent is willing to raise the last bid. The highest bidder wins the item at her proposed price (Nisan et al., 2007).

In this specification, let us consider two agents, with $\mathrm{N} = \{\mathsf{i}_1, \mathsf{i}_2\}$. We let $-\mathsf{i}$ denote the opponent of $\mathsf{i} \in \mathrm{N}$. Furthermore, we let $\Phi = \{\text{price}, \text{initial}, \text{choice}_\mathsf{i},$

$\mathrm{bid}_\mathsf{i}, \mathrm{turn}_\mathsf{i}, \mathrm{pay}_\mathsf{i}, \mathrm{terminal} : \mathsf{i} \in \mathrm{N}\}$, where price denotes the current price, initial denotes whether the position is the initial one, turn_i specifies whether it is i's turn and bid_i specifies the value of i's bid.

The following $\mathsf{SL}[\mathcal{F}]$-formulae are a partial description of a two-player turned-based variant of the English auction. Rule E1 says that proposition initial implies it is the turn of agent i_1, the position is not terminal and the bids are zero. Rule E2 defines the price as the highest bid among the two players. Rule E3 specifies the turn taking. Rules E4-E6 specify the choice of winner: when an agent outbids her opponent, she is the (temporary) winner. If she bids lower, the game ends and the opponent wins. Finally, if someone bids 1, the auction ends and the winner is the player who had the turn. The value of formula E7 is the social welfare in the best NE for type profile $\boldsymbol{\theta}$.

E1. $\mathbf{AG}(\mathrm{initial} \to \mathrm{turn}_{\mathsf{i}_1} \wedge \neg\mathrm{turn}_{\mathsf{i}_2} \wedge \neg\mathrm{terminal} \wedge \bigwedge_{\mathsf{i}\in\mathrm{N}}(\mathrm{bid}_\mathsf{i} = 0))$

E2. $\mathbf{AG}(\mathrm{price} = \max(\mathrm{bid}_{\mathsf{i}_1}, \mathrm{bid}_{\mathsf{i}_2}))$

E3. $\mathbf{AG}(\neg\mathrm{terminal} \to \bigwedge_{\mathsf{i}\in\mathrm{N}}(\mathrm{turn}_\mathsf{i} \to \neg\mathbf{X}\mathrm{turn}_\mathsf{i} \wedge \neg\mathrm{turn}_\mathsf{i} \to \mathbf{X}\mathrm{turn}_\mathsf{i}))$

E4. $\mathbf{AG}(\neg\mathrm{terminal} \to \bigwedge_{\mathsf{i}\in\mathrm{N}}(\mathrm{bid}_{-\mathsf{i}} < \mathbf{X}\mathrm{bid}_\mathsf{i} \wedge \mathrm{turn}_\mathsf{i} \to \mathbf{X}\mathrm{choice} = \mathrm{wins}_\mathsf{i}))$

E5. $\mathbf{AG}(\neg\mathrm{terminal} \to \bigwedge_{\mathsf{i}\in\mathrm{N}}(\mathrm{bid}_{-\mathsf{i}} \geq \mathbf{X}\mathrm{bid}_\mathsf{i} \wedge \mathrm{turn}_\mathsf{i} \to \mathbf{X}(\mathrm{choice} = \mathrm{wins}_{-\mathsf{i}}) \wedge \mathrm{terminal}))$

E6. $\mathbf{AG}(\bigwedge_{\mathsf{i}\in\mathrm{N}}(\mathrm{bid}_\mathsf{i} = 1 \wedge \mathrm{turn}_\mathsf{i} \wedge \mathbf{X}(\mathrm{choice} = \mathrm{wins}_\mathsf{i}) \wedge \mathrm{terminal}))$

E7. $\text{Max-Nash}(\sum_{\mathsf{i}\in\mathrm{N}} \mathsf{v}_\mathsf{i}(\mathrm{choice}, \theta_\mathsf{i}), \boldsymbol{\theta})$

Letting $\Sigma_{\mathrm{eng}}(\boldsymbol{\theta})$ be the conjunction of Rules E1-E7 alongside with the payment rules in Σ_{jpn} (Rules J7-J8), we have:

Proposition 7.3. *There exists a wCGS-mechanism $\mathcal{G}_{\mathrm{eng}}$ such that $[\![\Sigma_{\mathrm{eng}}(\boldsymbol{\theta})]\!]^{\mathcal{G}_{\mathrm{eng}}} = \max_{x\in\mathcal{X}} \sum_{\mathsf{i}\in\mathrm{N}} \mathsf{v}_\mathsf{i}(x, \theta_\mathsf{i})$, for each type profile $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.*

Proof. We construct a wCGS-mechanism $\mathcal{G}_{\mathrm{eng}}$ in which Σ_{eng} has the satisfaction value equal to one, for any history and assignment. Let $0 > \mathsf{inc} > 1$ be a constant value and the action set be defined as $\mathcal{B} = \{1 - x \cdot \mathsf{inc} : 0 \leq x \leq \frac{1}{\mathsf{inc}} \cup \{0\}$, where each action denotes the value the agent is willing to pay for the item. Let $\mathcal{G}_{\mathrm{eng}} = (\mathcal{B}, V, \delta, \ell, v_\iota)$ over $\mathrm{AP} = \{\mathrm{price}, \mathrm{initial}, \mathrm{choice}_\mathsf{i}, \mathrm{bid}_\mathsf{i}, \mathrm{turn}_\mathsf{i}, \mathrm{pay}_\mathsf{i}, \mathrm{terminal} : \mathsf{i} \in \mathrm{N}\}$, where:

- V consists of positions of the form $\langle(\mathrm{b}_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}, \mathrm{own}, \mathrm{winner}, \mathrm{tr}\rangle$ with $\mathrm{b}_\mathsf{i} \in \mathcal{B}$ denoting the value of i's bid and $\mathrm{own} \in \mathrm{N}$ specifying the owner of the position and $\mathrm{tr} \in \{-1, 1\}$ denoting whether the position is terminal;
- In an initial position, the bids are 0, i_1 has the turn and it is not a terminal position. That is, the initial position is $v_\iota = \{\langle(0, 0), \mathsf{i}_1, \mathsf{i}_1, -1\rangle\}$.
- For each position $v = \langle(\mathrm{b}_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}, \mathrm{own}, \mathrm{winner}, \mathrm{tr}\rangle$ and joint action $\boldsymbol{a} = (a_\mathsf{i})_{\mathsf{i}\in\mathrm{N}}$, transition $\delta(v, \boldsymbol{a})$ is defined as follows:

– If $\mathrm{tr} = -1$, $\delta(v, \boldsymbol{a}) = \langle (\mathrm{b}'_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}, \mathrm{own}', \mathrm{winner}, \mathrm{tr}' \rangle$ where:

$$\mathrm{b}'_{\mathsf{i}} = \begin{cases} a_{\mathsf{i}} & \text{if own} = \mathsf{i}e \\ \mathrm{b}_{\mathsf{i}} & \text{otherwise} \end{cases}$$

$$\mathrm{own}' = \begin{cases} \mathsf{i} & \text{if own} \neq \mathsf{i}e \\ -\mathsf{i} & \text{otherwise} \end{cases}$$

$$\mathrm{winner}' = \begin{cases} \mathsf{i} & \text{if } a_{\mathsf{i}} > \mathrm{b}_{-\mathsf{i}} \\ -\mathsf{i} & \text{otherwise} \end{cases}$$

$$\mathrm{tr}' = \begin{cases} 1 & \text{if } \max(a_{\mathrm{own}}, \mathrm{b}_{-\mathrm{own}}) = 1 \text{ or} \\ & a_{\mathrm{own}} \leq \mathrm{b}_{-\mathrm{own}} \\ -1 & \text{otherwise} \end{cases}$$

– Otherwise, $\delta(v, \boldsymbol{a}) = v$.

- For each $v = \langle (\mathrm{b}_{\mathsf{i}})_{\mathsf{i}\in\mathrm{N}}, \mathrm{own}, \mathrm{winner}, \mathrm{tr} \rangle$ and each $\mathsf{i} \in \mathrm{N}$, the weight function is defined as follows:

  – $\ell(v, \mathrm{price}) = \max(\mathrm{b}_{\mathsf{i}_1}, \mathrm{b}_{\mathsf{i}_2})$,

  – $\ell(v, \mathrm{initial}_{\mathsf{i}}) = 1$ iff $v = v_\iota$ and $\ell(v, \mathrm{initial}_{\mathsf{i}}) = -1$ otherwise,

  – $\ell(v, \mathrm{choice}) = \mathrm{wins}_{\mathrm{winner}}$,

  – $\ell(v, \mathrm{bid}_{\mathsf{i}}) = \mathrm{b}_{\mathsf{i}}$,

  – $\ell(v, \mathrm{turn}_{\mathsf{i}}) = 1$ iff $\mathrm{own} = \mathsf{i}$ and $\ell(v, \mathrm{turn}_{\mathsf{i}}) = -1$ otherwise,

  – $\ell(v, \mathrm{pay}_{\mathsf{i}}) = \ell(v, \mathrm{price})$ iff $\mathrm{wins}_{\mathsf{i}} = \mathrm{winner}$ and $\ell(v, \mathrm{pay}_{\mathsf{i}}) = 0$ otherwise, and

  – $\ell(v, \mathrm{terminal}_{\mathsf{i}}) = \mathrm{tr}$.

It is straightforward to see that Rules E1-E7 and Rules J7-J8 have the satisfaction value 1 for any history in Hist and assignment $\mathcal{A}$.

Let $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. The proof for Rule E7 proceeds similarly to the proof for Rule J9 (Proposition 7.2), by noticing the strategy profile in which the agents' raise their bid the minimum possible (up to their valuation) in each turn is a Nash equilibrium. Furthermore, since this equilibrium is efficient, Rule E7 will have the satisfaction value equal to the maximum social welfare, that is $[\![\Sigma_{\mathrm{eng}}(\boldsymbol{\theta})]\!]^{\mathcal{G}_{\mathrm{eng}}} = \max_{x\in\mathcal{X}} \sum_{\mathsf{i}\in\mathrm{N}} \mathsf{v}_{\mathsf{i}}(x, \theta_{\mathsf{i}})$. □

As a result, Algorithm 3 applied to $\bigwedge_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \Sigma_{\mathrm{eng}}(\boldsymbol{\theta})$ returns a mechanism that satisfies all the rules E1-E7 and J7-J8, and in which the minimal social welfare in all possible type profiles is as high as possible.

Complexity The synthesis problem in these examples can be solved in 3-EXPTIME. The complexity is dominated by Rules J9 and E7, which express the existence of NE. Without them the complexity would be in 2-EXPTIME. Most importantly, the complexity is only in the size of the formula, which is typically rather small.

Approximate mechanisms The well-known results of Green and Laffont (1979) and Myerson and Satterthwaite (1983) show the impossibility of defining mechanisms whose equilibrium is efficient while having strict balance of monetary transfers. These impossibility results motivate the design of mechanisms that attempt to circumvent this problem by approximating or relaxing target properties. The quantitative semantics of $\mathsf{SL}[\mathcal{F}]$ and Algorithm 3 enable synthesizing mechanisms that approximate efficiency by maximizing social welfare.

7.4 Conclusion

We propose a novel approach for Automated Mechanism Design in which mechanisms can be automatically generated (or *synthesized*) 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 of our work is the use of formal methods techniques, which are developed to guarantee their correctness by construction.

While mechanism synthesis from $\mathsf{SL}[\mathcal{F}]$ specifications is undecidable, we solve it in two cases: when the number of actions is bounded, and when agents play in turn. We achieve this thanks to reductions to the satisfiability problem for $\mathsf{BQCTL}^*[\mathcal{F}]$, which we prove to be decidable. These two restrictions still preserve enough expressiveness to model relevant scenarios of mechanism synthesis, which we illustrate with examples based on auctions. The high complexity of the synthesis problem is only in the size of the formula, which is typically rather small as we saw in the examples.

The mechanism generated by Algorithm 3 does not need to be unique. The algorithm returns an arbitrary mechanism that maximizes the satisfaction value of the specification. Choosing the *best* mechanism (*e.g.*, in terms of compactness of the wCGS) when the solution is not unique is an interesting open problem.

We also notice that some aspects of AMD could be expressed in SL with standard Boolean-valuated semantics. However, the quantitative semantics of $\mathsf{SL}[\mathcal{F}]$ makes it possible to synthesise mechanisms that approximate a specification (satisfy it as much as possible), or maximize some value (such as social welfare in an equilibrium), which is not possible with SL.

Chapter 8

Conclusion

This thesis investigated an application of logics and strategic reasoning for Game Theory and Multi-Agent Systems. In particular, we propose the use of formal methods for the specification, design and evaluation of auctions-based markets and, more generally, preference aggregation mechanisms. In this final chapter, we summarize the contribution of each chapter and discuss perspectives for future work.

8.1 Summary of Contributions and Discussion

The first part of this thesis handled the problem of representing auctions using logical languages. In Chapter 2, we have presented the Auction Description Language (ADL), a unified framework for representing auction protocols. Our work is at the frontier of auction theory and knowledge representation. ADL provides tools for automated verification of properties for direct mechanisms. We showed how stages in a state transition model may represent direct mechanisms and be evaluated as such. The majority of properties that we considered (noting that at the evaluating a stage in an ST-model essentially boils down to model-checking ADL-formulae) can be checked in Ptime when the functions considered can be computed in polynomial time. Thus, ADL enables reasoning about important aspects for designing and playing auctions, while having a reasonable complexity cost.

ADL addresses important dimensions of auction-based markets and is general enough to represent most auction settings. In Chapter 3, we illustrated its usefulness by showing how to represent a number of representative auctions, which include features from single and multi-stage protocols, multiple items, multiple copies of those items and exchange protocols (which generalize double-sided auctions). We evaluate such protocols in relation to the well-formedness of their descriptions and the aforementioned properties of direct mechanisms.

In Chapter 4, we extended ADL to allow reasoning about knowledge and action choice. The resulting language (denoted ADLK), includes epistemic operators and action modalities and is aimed for the design of General Auction Players and the characterization of their rational behavior when reasoning about actions and other players' rationality. Since real world players may have time restrictions to decide their actions, we explore bounded rationality in relation to the level of higher-order knowledge about other agents and bounded looking-ahead beyond the next state. However, the inclusion of new operators came with a computational cost, as the model-checking problem for ADLK is in Exptime.

The main limitation of ADL is the semantics based on fixed paths, that is, non-alternating executions of an auction. This means it is not possible to encode through ADL-formulae conditions that compare the effect of different strategic behavior. An example of such condition is strategyproofness, where one should contrast the agents' utility after truthfully reporting her preference and after lying. Here, we demonstrated the use of meta-reasoning over the state-transition model for comparing alternative executions of an auction. Furthermore, evaluating indirect mechanisms requires capturing the terminal outcomes (that is, the final trades and payments) in strategic equilibrium. Complex solution concepts, such as Nash and dominant strategy equilibrium, cannot be encoded through ADL-formulae. Logics focused on strategic reasoning are more suitable for considering this problem (*e.g.*, ATL with Strategy Contexts (Brihaye et al., 2009) and SL (Chatterjee et al., 2010)). Finding a balance between expressivity and complexity is an open question, as such expressive languages face decidability issues and high complexity for model-checking (*e.g.*, the satisfiability problem of SL is undecidable and its complexity for model-checking is NonElementary).

The second part of this thesis investigated strategic reasoning and formal methods for Automated Mechanism Design (AMD). Chapter 5 proposes the use of a strategic logic for the verification of mechanisms. The logic considered is a new variant of Strategy Logic with quantitative features, imperfect information and epistemic operators, that we called SLK[$\mathcal{F}$]. We first showed how mechanisms can be cast as concurrent-game structures. We then showed how SLK[$\mathcal{F}$] can express that a mechanism implements a social choice function, a fundamental concept in mechanism design. This then allowed us to express in SLK[$\mathcal{F}$] whether a mechanism satisfies desired properties related to the expected behavior of the participants as well as to the quality of the chosen outcome. We illustrated this with a number of important properties often required in auctions, or more generally in mechanism design: strategyproofness, individual rationality, efficiency, budget-balance and Pareto optimality (Nisan et al., 2007). We also considered epistemic aspects and showed how, thanks to the epistemic operators in 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 of model checking an SLK[$\mathcal{F}$] formula, which we show can be done in Pspace for memoryless strategies (and is thus a Pspace-complete problem).

Still in the context of formal verification of auctions and mechanisms, in Chapter 6 we consider the problem of reasoning about strategies that are human-readable while also being machine processable. We introduced a quantitative semantics for SL with Natural Strategies and imperfect information (denoted NatSL[$\mathcal{F}$]) and argued it provides a new perspective for the verification and design of novel mechanisms based on the complexity of strategies. First, we focused our attention on an interesting type of repeated mechanism: the keyword ad auction. We show how to model popular strategies for this game using NatSL[$\mathcal{F}$] and proved properties pertaining to this game. In a second stage, we analysed our novel variant of SL in relation with its distinguishing power, expressivity, and complexity of the model

checking problem, for natural strategies with and without recall. We proved that the model-checking problem for $\mathsf{NatSL}[\mathcal{F}]$ is PSPACE-complete and that $\mathsf{NatSL}[\mathcal{F}]$ has incomparable distinguishing and expressive power to $\mathsf{SL}[\mathcal{F}]$ with standard combinatorial strategies of arbitrary complexity. This means that $\mathsf{NatSL}[\mathcal{F}]$ allows to express properties that cannot be captured in $\mathsf{SL}[\mathcal{F}]$ (and vice-versa).

Finally, in Chapter 7 we proposed a new approach for AMD, which offers a novel perspective on the design of mechanisms. Our approach enables automatically generating optimal mechanisms from a quantitative logical specification, which may include not only game rules but also requirements over the strategic behavior of participants and quality of the outcome. We rephrased the AMD problem in terms of synthesis from $\mathsf{SL}[\mathcal{F}]$ specifications. To solve this synthesis problem we investigated the related satisfiability problem for $\mathsf{SL}[\mathcal{F}]$, which had not been studied so far. We show that, as for classic SL (Laroussinie and Markey, 2015) the problem is undecidable but can be solved in two cases: when the number of actions is bounded, and when agents play in turn. In both cases, we show that the complexity is not worse than the standard Boolean-valuated SL. We then illustrated the relevance of mechanism synthesis in these two cases with examples based on Auction Design. An advantage of this approach is that it can be used to approximate target properties and therefore minimize the effects entailed by economic impossibility results.

8.2 Perspectives and Future Work

There are many directions from this work that we believe would be worth investigating in the future. In most of this thesis, we put the emphasis on the auctioneer and mechanism designer. A first direction is to design a ADL-based general auction player that can interpret and reason about the rules of an auction-based market. In such case, search optimization techniques used for General Game Playing (see, for instance (Finnsson, 2012; Wang et al., 2018)) may be adapted for considering utility optimization in auctions.

It is also an interesting line of work to develop the axiomatic system for ADL and prove its soundness and completeness with respect to the semantics based on state transition models. It would require a combination of techniques used for Epistemic GDL (Jiang et al., 2016) and first-order logic with dependent types (Rabe, 2006).

In relation to AMD, the use of the probabilistic extension of SL (Aminof et al., 2019) would allow handling stochastic features often present in auctions. Going from deterministic setting to a more general and probabilistic one is challenging due to several aspects. First, the wide and heterogeneous range of settings considered in the literature obscures the path for a general and formal approach to verification. The setting may consider deterministic or randomized mechanisms, incomplete information about agents' types (Bayesian mechanisms), mixed or pure strategies and iterative protocols (indirect mechanisms). Second, considering Bayesian mechanisms brings out different methods for evaluating a mechanism according to the time-line for revealing the incomplete information as the game is run.

We studied the verification of mechanisms under memoryless combinatorial strategies and Natural Strategies with bounded recall. This setting is enough to capture many kinds of auctions (such as one-shot or English auctions) where memoryless strategies are sufficient to represent the bidders' behaviour since all the relevant information can be encoded in a state. However, when participating in repeated auctions, agents could gather information from other agents' behaviour and act based on what happened in previous instances of the game. An interesting direction is, then, to investigate the use of strategies with recall for learning other players' valuations based on their behavior. For such situations we can 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), which should be considered also in the quantitative case.

We believe the automated synthesis of mechanisms is a promising and powerful tool for AMD. However, the high expressiveness of $\mathsf{SL}[\mathcal{F}]$ may not always be needed for simple classes of mechanisms, and one may consider fragments of it to achieve better complexity. Therefore, an interesting direction for future work is to study the complexity of synthesizing from $\mathsf{SL}[\mathcal{F}]$-fragments, inspired from the SL-fragments One-Goal SL (Mogavero et al., 2017; Cermák et al., 2015) and Simple-Goal SL (Belardinelli et al., 2019), for instance. These fragments are usually computationally easier than full SL, and we can hope that similar results can be established in the quantitative setting. On a related vein, we can study the translation of ADL to $\mathsf{SL}[\mathcal{F}]$-formulae, so as to include the auction description in the mechanism specification. In this setting, $\mathsf{SL}[\mathcal{F}]$ formulae can be used to express requirements of well-formed auction descriptions.

The problems contemplated in this thesis are also worth investigating from an empirical perspective. One direction is to explore the interplay between agents' bounded rationality and the auctioneer revenue so as to understand the impact of bounded rationality on mechanism design. An implementation of a model checker for $\mathsf{NatSL}[\mathcal{F}]$ would enable the empirical evaluation of natural strategies and auctions played by participants with restricted memory. Finally, experimental results could be used to assess the practical relevance of our proposed approaches, especially in relation to mechanism synthesis from $\mathsf{SL}[\mathcal{F}]$ specification due to the high theoretical complexity of the problem.

Appendix A

Complexity classes

Let us recall some classical notations in the theory of computational complexity. For all $k, n \in \mathbb{N}$, we first define the iterated exponential functions $\exp^0(n) = n$ and $\exp^{k+1}(n) = 2^{\exp^k}(n)$.

By Ptime (respectively, NP) we denote the class of languages (*i.e.*, decision problems) decidable in polynomial-time deterministic (respectively, nondeterministic) Turing machines. Δ_2^{P} denotes the class of languages each of which is decidable in a polynomial-time deterministic Turing machine with a polynomial number of queries to an NP language as an oracle.

We note Pspace (respectively, Expspace) for the class of problems solvable in polynomial space (respectively, exponential space), and for each $k \in \mathbb{N}$, we let k-Exptime be the class of problems that can be solved in time $\exp^{k+1}(n)$ for some constant $c \in \mathbb{N}$ (where n is the size of the input).

We also define the class Elementary of elementary problems, where:

$$\textsc{Elementary} = \bigcup_{k \in \mathbb{N}} k\text{-}\textsc{Exptime}$$

If a problem is decidable but not elementary it is NonElementary.

Appendix B

Satisfiability of $\mathsf{BQCTL}^*[\mathcal{F}]$

We now present the full proof for the satisfiability of $\mathsf{BQCTL}^*[\mathcal{F}]$.

Additional definitions

A tree $t = (\tau, \ell)$ is *complete* if for all node $u \in \tau$ and direction $d \in X$, we have $u \cdot d \in \tau$. Given a $\mathcal{V}^{\mathrm{AP}}$-labeled X-tree $t = (\tau, \ell)$, we let $\bar{t} = (\overline{\tau}, \overline{\ell})$ be the only $\mathcal{V}^{\mathrm{AP}} \cup \{\bullet\}$-labeled X-tree such that for all $u \in \tau$, $\overline{\ell}(u) = \ell(u)$, and for all $u \in \overline{\tau} \setminus \tau$, $\overline{\ell}(u) = \{\bullet\}$, where $\bullet$ is a fresh symbol that labels artificial nodes added to make the tree complete. Similarly, every $\mathcal{V}^{\mathrm{AP}} \cup \{\bullet\}$-tree $\bar{t}$ induces a unique $\mathcal{V}^{\mathrm{AP}}$-tree t obtained by removing each subtree rooted in a node labeled with $\bullet$.

We refer the reader to (Pin, 2021, Chapter 8) for standard definitions and results for automata on infinite trees.

Reduction to binary branching

We first show that the satisfiability problem for $\mathsf{BQCTL}^*[\mathcal{F}]$ in finite structures can be reduced to the same problem restricted to structures with binary branching. The proof is a simple adaptation to the quantitative setting of the one in (Laroussinie and Markey, 2014) for QCTL^*. We sketch the proof, and refer the reader to (Laroussinie and Markey, 2014) for details.

For every wKS $\mathcal{S}$ we can define a wKS $\widetilde{\mathcal{S}}$ with binary branching that simulates wKS: for each state of $\mathcal{S}$ with k successors, $\widetilde{\mathcal{S}}$ contains a binary tree with k leaves, one for each successor. Internal nodes are dummy nodes, labelled with a fresh atomic proposition p_{int}. More precisely, p_{int} has value 1 in internal nodes, and -1 in normal states of $\mathcal{S}$.

Now for every formula $\varphi \in \mathsf{BQCTL}^*[\mathcal{F}]$ we can define inductively a formula $\widetilde{\varphi}$ such that $[\![\varphi]\!]^{\mathcal{S}} = [\![\widetilde{\varphi}]\!]^{\widetilde{\mathcal{S}}}$. We only give the inductive case for temporal operators, all other cases distribute over the operators:

$$\widetilde{\mathbf{X}\varphi} = \mathbf{X}\,[p_{\mathrm{int}}\mathbf{U}(\neg p_{\mathrm{int}} \wedge \widetilde{\varphi})]$$
$$\widetilde{\varphi\mathbf{U}\varphi'} = (p_{\mathrm{int}} \vee \widetilde{\varphi})\mathbf{U}(\neg p_{\mathrm{int}} \wedge \widetilde{\varphi'})$$

We have the following:

Lemma B.1. *For every* $\mathsf{BQCTL}^*[\mathcal{F}]$ *formula* φ *and every finite wKS* $\mathcal{S}$, $[\![\varphi]\!]^{\mathcal{S}} = [\![\widetilde{\varphi}]\!]^{\widetilde{\mathcal{S}}}$.

Deciding satisfiability of $\mathsf{BQCTL}^*[\mathcal{F}]$

Let $\mathrm{nd}(\varphi)$ be the maximal number of nested quantifiers on propositions in φ. The following result was established in (Bouyer et al., 2019), for the tree semantics of $\mathsf{BQCTL}^*[\mathcal{F}]$:

Proposition B.1. *Let $\mathcal{V} \subset [-1,1]$ be a finite set of values such that $\{-1,1\} \subseteq \mathcal{V}$, and let D be a finite set of directions. For every $\mathsf{BQCTL}^*[\mathcal{F}]$ state formula φ and predicate $P \subseteq (-1,1]$, one can construct an APT $\mathcal{A}_\varphi^{\mathcal{V},P}$ over $(\mathcal{V}^{\mathrm{AP}} \cup \{\bullet\})$-trees such that for every $\mathcal{V}^{\mathrm{AP}}$-labeled D-tree t,*

$$\mathcal{A}_\varphi^{\mathcal{V},P} \text{ accepts } \bar{t} \text{ if and only if } [\![\varphi]\!]^t \in P.$$

The APT $\mathcal{A}_\varphi^{\mathcal{V},P}$ is of size at most $(\mathrm{nd}(\varphi)+1)$-exponential in $|\varphi|$, and its index is at most $\mathrm{nd}(\varphi)$-exponential in $|\varphi|$.

From this result we obtain that satisfiability of $\mathsf{BQCTL}^*[\mathcal{F}]$ formulas is decidable.

Theorem B.1. *The satisfiability problem for $\mathsf{BQCTL}^*[\mathcal{F}]$ is nonelementary decidable. For formulas of nesting depth at most k, the problem is $(k+1)$-*Exptime*-complete.*

Proof. The lower bounds are inherited from the satisfiability problem for QCTL^* (Laroussinie and Markey, 2014). The reduction is direct with threshold $\varepsilon = 1$.

For membership, let Φ be a $\mathsf{BQCTL}^*[\mathcal{F}]$ formula and $\varepsilon > -1$ a threshold. Let $\widetilde{\Phi}$ be the corresponding formula on structures with binary branching, and let $\varphi_2 = \neg p_{\mathrm{int}} \wedge \mathbf{AGF}\neg p_{\mathrm{int}}$.

Lemma B.2. *There exists a (finite) wKS $\mathcal{S}$ such that $[\![\Phi]\!]^{\mathcal{S}} \geq \varepsilon$ if and only if there exists a regular binary tree t such that $[\![\mathbf{AG}\mathrm{Bool}(p_{\mathrm{int}}) \wedge \varphi_2 \wedge \widetilde{\Phi}]\!]^t \geq \varepsilon$.*

Proof. If there exists $\mathcal{S}$ such that $[\![\Phi]\!]^{\mathcal{S}} \geq \varepsilon$, then by Lemma B.1 $[\![\widetilde{\Phi}]\!]^{\widetilde{\mathcal{S}}} \geq \varepsilon$. Also by construction we have $[\![\mathbf{AG}\mathrm{Bool}(p_{\mathrm{int}})]\!]^{\widetilde{\mathcal{S}}} = 1$ and $[\![\varphi_2]\!]^{\widetilde{\mathcal{S}}} = 1$, so that $[\![\mathbf{AG}\mathrm{Bool}(p_{\mathrm{int}}) \wedge \varphi_2 \wedge \widetilde{\Phi}]\!]^t \geq \varepsilon$, where $t = t_{\mathcal{S}}$ is the regular tree obtained by unfolding $\widetilde{\mathcal{S}}$.

For the other direction assume there exists a regular binary tree t such that $[\![\mathbf{AG}\mathrm{Bool}(p_{\mathrm{int}}) \wedge \varphi_2 \wedge \widetilde{\Phi}]\!]^t \geq \varepsilon$. There exists a wKS $\mathcal{S}'$ with binary branching of which t is the unfolding. Since $\varepsilon > -1$ and $\mathbf{AG}\mathrm{Bool}(p_{\mathrm{int}}) \wedge \varphi_2$ can only take values 1 or -1, we have that $[\![\mathbf{AG}\mathrm{Bool}(p_{\mathrm{int}}) \wedge \varphi_2]\!]^{\mathcal{S}'} = 1$. It follows that $[\![\widetilde{\Phi}]\!]^{\mathcal{S}'} \geq \varepsilon$ and that $\mathcal{S}' = \widetilde{\mathcal{S}}$ for some $\mathcal{S}$. By Lemma B.1 we have that $[\![\Phi]\!]^{\mathcal{S}} = [\![\widetilde{\Phi}]\!]^{\widetilde{\mathcal{S}}} \geq \varepsilon$. □

Now from Proposition B.1 with $P = [\varepsilon, 1]$ and $\varphi = \mathbf{AG}\mathrm{Bool}(p_{\mathrm{int}}) \wedge \varphi_2 \wedge \widetilde{\Phi}$ we can build an automaton $\mathcal{A}$ over $(\mathcal{V}^{\mathrm{AP}} \cup \{\bullet\})$-labeled binary trees such that for every $\mathcal{V}^{\mathrm{AP}}$-labeled binary tree t, $\mathcal{A}$ accepts $\bar{t}$ if and only if $[\![\Phi]\!]^t \geq \varepsilon$. From Lemma B.2 and the fact that every regular tree language contains a regular tree it follows that $\mathcal{A}$ is nonempty if and only if there exists a finite wKS $\mathcal{S}$ such that $[\![\Phi]\!]^t \geq \varepsilon$. The emptiness of $\mathcal{A}$ can be tested in time polynomial in the number of states and

exponential in the index (Löding, 2021). The overall complexity is thus $(k+1)$-exponential in time. □

Actually, observing the construction in (Bouyer et al., 2019) one sees that sequences of consecutive quantifiers on propositions can be treated in only one exponential, if they are all existential or all universal.

Appendix C

Published work

The work presented in this thesis is based on the following publications:

- Chapter 2 and 3 are based on:
  - Mittelmann, M. and Perrussel, L. (2020c). Game description logic with integers: A GDL numerical extension. In *Proceedings of the International Symposium on Foundations of Information and Knowledge Systems (FoIKS 2020)*, volume 12012, pages 191–210. Springer
  - Mittelmann, M. and Perrussel, L. (2020a). Auction description language (ADL): general framework for representing auction-based markets. In *Proceedings of the European Conference on Artificial Intelligence (ECAI 2020)*, volume 325, pages 825–832. IOS Press
  - Mittelmann, M., Bouveret, S., and Perrussel, L. (2021a). A general framework for the logical representation of combinatorial exchange protocols (extended abstract). In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2021)*, pages 1602–1604. ACM
  - Mittelmann, M., Bouveret, S., and Perrussel, L. (2022a). Representing and reasoning about auctions. *Autonomous Agents and Multi-Agent Systems*, 36(1):20
- Chapter 4 is based on:
  - Mittelmann, M. and Perrussel, L. (2020b). An epistemic logic for reasoning about strategies in general auctions. In *Proceedings of the Workshops of the International Conference on Logic Programming*, volume 2678
  - Mittelmann, M., Herzig, A., and Perrussel, L. (2021b). Epistemic reasoning about rationality and bids in auctions. In *Proceedings of the European Conference on Logics in Artificial Intelligence (JELIA 2021)*, volume 12678, pages 116–130. Springer
- Chapter 5 is based on:
  - Maubert, B., Mittelmann, M., Murano, A., and Perrussel, L. (2021). Strategic reasoning in automated mechanism design. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR 2021)*, pages 487–496

- Chapter 6 is based on:
  - Belardinelli, F., Jamroga, W., Malvone, V., Mittelmann, M., Murano, A., and Perrussel, L. (2022). Reasoning about human-friendly strategies in repeated keyword auctions. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2022)*, pages 1602–1604. IFAAMAS
- Chapter 7 is based on:
  - Mittelmann, M., Maubert, B., Murano, A., and Perrussel, L. (2022b). Automated synthesis of mechanisms. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2022) - to appear*

Bibliography

Ågotnes, T. (2006). Action and knowledge in alternating-time temporal logic. *Synthese*, 149(2):377–409.

Ågotnes, T., Van Der Hoek, W., Rodríguez-Aguilar, J. A., Sierra, C., and Wooldridge, M. J. (2007). On the logic of normative systems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2007)*, volume 7, pages 1175–1180.

Ågotnes, T. and Walther, D. (2009). A logic of strategic ability under bounded memory. *Journal of Logic, Language and Information*, 18(1):55–77.

Alechina, N., Dastani, M., Logan, B., and Meyer, J.-J. C. (2007). A logic of agent programs. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2007)*, pages 795–800, Vancouver. AAAI Press.

Alechina, N., Logan, B., Dastani, M., and Meyer, J.-J. C. (2008). Reasoning about agent execution strategies. In *Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, pages 1455–1458, Estoril. IFAAMAS.

Alechina, N., Logan, B., Nga, N., and Rakib, A. (2009). A logic for coalitions with bounded resources. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pages 659–664, Pasadena. AAAI Press.

Alechina, N., Logan, B., Nguyen, H., and Rakib, A. (2010). Resource-bounded alternating-time temporal logic. In *Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 481–488, Toronto. IFAAMAS.

Almagor, S., Boker, U., and Kupferman, O. (2016). Formally reasoning about quality. *Journal of the ACM*, 63(3):24:1–24:56.

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

Aminof, B., Kwiatkowska, M., Maubert, B., Murano, A., and Rubin, S. (2019). Probabilistic strategy logic. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI 2019)*.

Asselin, F., Jaumard, B., and Nongaillard, A. (2006). A technique for large automated mechanism design problems. In *Proceedings of the International Conference on Intelligent Agent Technology (IAT 2006)*.

Aumann, R. (1995). Backward induction and common knowledge of rationality. *Games and Economic Behavior*, 8:6—19.

Bădică, C., Ganzha, M., and Paprzycki, M. (2006). Rule-based automated price negotiation: Overview and experiment. In *Artificial Intelligence and Soft Computing (ICAISC 2006)*, pages 1050–1059, Berlin, Heidelberg. Springer Berlin Heidelberg.

Baier, C. and Katoen, J.-P. (2008). *Principles of Model Checking (Representation and Mind Series)*. The MIT Press, Cambridge.

Baral, C. and Uyan, C. (2001). Declarative specification and solution of combinatorial auctions using logic programming. In *Logic Programming and Nonmotonic Reasoning*, pages 186–199, Berlin, Heidelberg. Springer.

Barlo, M., Carmona, G., and Sabourian, H. (2008). Bounded memory with finite action spaces. *Sabanci University, Universidade Nova de Lisboa and University of Cambridge*, 1(1).

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

Belardinelli, F., Jamroga, W., Kurpiewski, D., Malvone, V., and Murano, A. (2019). Strategy logic with simple goals: Tractable reasoning about strategies. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2019)*.

Belardinelli, F., Jamroga, W., Malvone, V., Mittelmann, M., Murano, A., and Perrussel, L. (2022). Reasoning about human-friendly strategies in repeated keyword auctions. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2022)*, pages 1602–1604. IFAAMAS.

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

Belardinelli, F., Lomuscio, A., and Malvone, V. (2018). Approximating perfect recall when model checking strategic abilities. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR 2018)*, pages 435–444. AAAI Press.

Belardinelli, F., Lomuscio, A., Murano, A., and Rubin, S. (2020). Verification of multi-agent systems with public actions against strategy logic. *Artificial Intelligence*, 285.

Belardinelli, F., Lomuscio, A., and Patrizi, F. (2014). Verification of agent-based artifact systems. *Journal of Artificial Intelligence Research*, 51:333–376.

Bellosta, M. ., Kornman, S., and Vanderpooten, D. (2005). A framework for multiple criteria english reverse auctions. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 633–639.

Bellosta, M.-J., Kornman, S., and Vanderpooten, D. (2008). A unified framework for multiple criteria auction mechanisms. *Web Intelligence and Agent Systems*, 6(4):401–419.

Belnap, N. and Perloff, M. (1990). Seeing to it that: A canonical form for agentives. In *Knowledge representation and defeasible reasoning*, pages 167–190. Springer.

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

Bonanno, G. (2015). Epistemic foundations of game theory. In H. van Ditmarsch, J.Y.Halpern, W. v. d. H. and Kooi, B., editors, *Handbook of Logics for Knowledge and Belief*, chapter 9, pages 411—450. College Publications.

Bordini, R., Fisher, M., Visser, W., and Wooldridge, M. (2006). Verifying multi-agent programs by model checking. *Autonomous Agents and Multi-Agent Systems*, 12(2):239–256.

Boutilier, C. and Hoos, H. H. (2001). Bidding languages for combinatorial auctions. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pages 1211–1217. Morgan Kaufmann.

Bouveret, S. and Lang, J. (2014). Manipulating picking sequences. In *Proceedings of the European Conference on Artificial Intelligence (ECAI 2014)*.

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

Brihaye, T., Da Costa, A., Laroussinie, F., and Markey, N. (2009). ATL with strategy contexts and bounded memory. In *International Symposium on Logical Foundations of Computer Science*, pages 92–106. Springer.

Bulling, N. and Dastani, M. (2016). Norm-based mechanism design. *Artificial Intelligence*, 239:97–142.

Bulling, N. and Farwer, B. (2010a). Expressing properties of resource-bounded systems: The logics RTL* and RTL. In *Proceedings of Computational Logic in Multi-Agent Systems (CLIMA 2010)*, page 22–45, Berlin. Springer-Verlag.

Bulling, N. and Farwer, B. (2010b). On the (un-)decidability of model checking resource-bounded agents. In *Proceedings of the European Conference on Artificial Intelligence (ECAI 2010)*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, pages 567–572, Amsterdam. IOS Press.

Bulling, N. and Jamroga, W. (2014). Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *Journal of Autonomous Agents and Multi-Agent Systems*, 28(3):474–518.

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

Cary, M., Das, A., Edelman, B., Giotis, I., Heimerl, K., Karlin, A. R., Mathieu, C., and Schwarz, M. (2007). Greedy bidding strategies for keyword auctions. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC 2007)*, page 262–271, New York. Association for Computing Machinery.

Cavallo, R., Parkes, D. C., Juda, A. I., Kirsch, A., Kulesza, A., Lahaie, S., Lubin, B., Michael, L., and Shneidman, J. (2005). TBBL: A tree-based bidding language for iterative combinatorial exchanges. In *Multidisciplinary Workshop on Advances in Preference Handling*, Edinburgh.

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

Cermák, P., Lomuscio, A., and Murano, A. (2015). Verifying and synthesising multi-agent systems against one-goal strategy logic specifications. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2015)*.

Chatterjee, K., Henzinger, T. A., and Piterman, N. (2010). Strategy logic. *Information and Computation*, 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).

Chen, J., Micali, S., and Pass, R. (2015). Tight revenue bounds with possibilistic beliefs and level-k rationality. *Econometrica*, 83(4):1619–1639.

Chevaleyre, Y., Dunne, P. E., Endriss, U., Lang, J., Lemaître, M., Maudet, N., Padget, J., Phelps, S., Rodríguez-Aguilar, J. A., and Sousa, P. (2006). Issues in Multiagent Resource Allocation. *Informatica*, 30(1):3–31.

Clarke, E., Grumberg, O., Kroening, D., Peled, D., and Veith, H. (2018). *Model checking*. MIT press.

Conitzer, V. and Sandholm, T. (2002). Complexity of mechanism design. In *Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence (UAI 2002), University of Alberta, Edmonton*, pages 103–110. Morgan Kaufmann.

Cramton, P. (2011). *Simultaneous Ascending Auctions*, chapter 4. American Cancer Society.

Dastani, M. and Jamroga, W. (2010). Reasoning about strategies of multi-agent programs. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2010)*, pages 625–632, Richland. International Foundation for Autonomous Agents and Multiagent Systems.

David, C. and Kroening, D. (2017). Program synthesis: challenges and opportunities. *Philosophical Transactions of the Royal Society A*, 375(2104):20150403.

de Jonge, D. and Zhang, D. (2021). GDL as a unifying domain description language for declarative automated negotiation. *Autonomous Agents and Multi-Agent Systems*, 35(1):13.

Dima, C. and Tiplea, F. (2011). Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*, abs/1102.4225.

Dobriceanu, A., Biscu, L., and Badica, C. (2007). Adding a declarative representation of negotiation mechanisms to an agent-based negotiation service. In *2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*, pages 471–474.

Duijf, H. and Broersen, J. (2016). Representing strategies. In *Proceedings of International Workshop on Strategic Reasoning (SR)*, pages 15–26, New York. Open Publishing Association.

Dütting, P., Feng, Z., Narasimhan, H., Parkes, D., and Ravindranath, S. S. (2019). Optimal auctions through deep learning. In *Proceedings of the International Conference on Machine Learning (ICML 2019)*.

Edelman, B., Ostrovsky, M., and Schwarz, M. (2007). Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97(1):242–259.

Fagin, R., Moses, Y., Halpern, J. Y., and Vardi, M. Y. (2003). *Reasoning about knowledge*. MIT press.

Feige, U., Feldman, M., Immorlica, N., Izsak, R., Lucier, B., and Syrgkanis, V. (2015). A unifying hierarchy of valuations with complements and substitutes. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2015)*, 29(1).

Finnsson, H. (2012). Generalized monte-carlo tree search extensions for general game playing. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI 2012)*.

French, T. (2003). Quantified propositional temporal logic with repeating states. In *Proceedings of the 10th International Symposium on Temporal Representation*

and Reasoning, 2003 and Fourth International Conference on Temporal Logic (TIME 2003).

Galal, H. S. and Youssef, A. M. (2019). Verifiable sealed-bid auction on the ethereum blockchain. In *Financial Cryptography and Data Security*, pages 265–278, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

Giovannucci, A., Cerquides, J., Endriss, U., and Rodríguez-Aguilar, J. A. (2010). A graphical formalism for mixed multi-unit combinatorial auctions. *Autonomous Agents and Multi-Agent Systems*, 20(3):342–368.

Green, J. and Laffont, J.-J. (1979). *Incentives in public decision-making*. Elsevier North-Holland.

Gupta, A., Schewe, S., and Wojtczak, D. (2015). Making the best of limited memory in multi-player discounted sum games. *Electronic Proceedings in Theoretical Computer Science*, 193:16–30.

Gutierrez, J., Najib, M., Perelli, G., and Wooldridge, M. J. (2019). Equilibrium design for concurrent games. In *30th International Conference on Concurrency Theory (CONCUR 2019)*.

Harel, D. and Kozen, D. (1982). Process logic: Expressiveness, decidability, completeness. *Journal of Computer and System Sciences*, 25(2):144–170.

Harrenstein, P., van der Hoek, W., Meyer, J.-J., and Witteveen, C. (2001). Boolean games. In *Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 287–298.

Herzig, A., Lorini, E., Maffre, F., and Schwarzentruber, F. (2016). Epistemic boolean games based on a logic of visibility and control. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 1116–1122.

Herzig, A., Lorini, E., Maffre, F., and Walther, D. (2014). Alternating-time temporal logic with explicit programs. In *Proceedings of Workshop on Logical Aspects of Multi-Agent Systems (LAMAS)*, Paris. IFAAMAS.

Hörner, J. and Olszewski, W. (2009). How robust is the folk theorem? *The Quarterly Journal of Economics*, 124(4):1773–1814.

Hudert, S., Eymann, T., Ludwig, H., and Wirtz, G. (2009a). A negotiation protocol description language for automated service level agreement negotiations. *2009 IEEE Conference on Commerce and Enterprise Computing (CEC 2009)*, pages 162–169.

Hudert, S., Ludwig, H., and Wirtz, G. (2009b). Negotiating SLAs-An approach for a generic negotiation framework for WS-agreement. *Journal of Grid Computing*, 7(2):225–246.

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

Jamroga, W. and Ågotnes, T. (2007). Constructive knowledge: what agents can achieve under imperfect information. *J. Applied Non-Classical Logics*, 17(4):423–475.

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

Jamroga, W., Kurpiewski, D., and Malvone, V. (2020). Natural strategic abilities in voting protocols. In *Socio-Technical Aspects in Security and Trust - 10th International Workshop (STAST 2020)*, volume 12812 of *Lecture Notes in Computer Science*, pages 45–62, Berlin. Springer.

Jamroga, W., Malvone, V., and Murano, A. (2019a). Natural strategic ability. *Artificial Intelligence*, 277:103170.

Jamroga, W., Malvone, V., and Murano, A. (2019b). Natural strategic ability under imperfect information. In *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2019)*, page 962–970, Richland. International Foundation for Autonomous Agents and Multiagent Systems.

Jamroga, W. and van der Hoek, W. (2004). Agents that know how to play. *Fundamenta Informaticae*, 63(2-3):185–219.

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

Jiang, G., Perrussel, L., and Zhang, D. (2017). On axiomatization of epistemic gdl. In *International Workshop on Logic, Rationality and Interaction*, pages 598–613, Berlin, Heidelberg. Springer.

Jiang, G., Perrussel, L., Zhang, D., Zhang, H., and Zhang, Y. (2019). Game equivalence and bisimulation for game description language. In *PRICAI 2019: Trends in Artificial Intelligence - 16th Pacific Rim International Conference on Artificial Intelligence*, volume 11670, pages 583–596. Springer.

Jiang, G., Zhang, D., Perrussel, L., and Zhang, H. (2016). Epistemic GDL: A logic for representing and reasoning about imperfect information games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2016)*.

Jiang, G., Zhang, D., Perrussel, L., and Zhang, H. (2021). Epistemic GDL: A logic for representing and reasoning about imperfect information games. *Artificial Intelligence*, 294:103453.

Jonge, D. d. and Zhang, D. (2016). Using GDL to represent domain knowledge for automated negotiations. In *Autonomous Agents and Multi-Agent Systems: AAMAS 2016 Workshops, Visionary Papers, Singapore*, pages 134–153, Cham. Springer International Publishing.

Jonge, D. d. and Zhang, D. (2017). Automated negotiations for general game playing. In *Proceedings of the 16th Conference on Autonomous Agents and Multi-Agent Systems, (AAMAS 2017), São Paulo*, pages 371–379. ACM.

Kalagnanam, J. and Parkes, D. C. (2004). Auctions, bidding and exchange design. In *Handbook of Quantitative Supply Chain Analysis: Modeling in the E-Business Era*, pages 143–212, Boston, MA. Springer US.

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

Klemperer, P. (1999). Auction theory: A guide to the literature. *Journal of Economic Surveys*, 13(3):227–286.

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

Kupferman, O. and Vardi, M. Y. (2000). Synthesis with incomplete informatio. In *Advances in Temporal Logic*, pages 109–127. Springer, Berlin.

Laroussinie, F. and Markey, N. (2014). Quantified CTL: expressiveness and complexity. *Logical Methods in Computer Science*, 10(4).

Laroussinie, F. and Markey, N. (2015). Augmenting ATL with strategy contexts. *Information and Computation*, 245:98–123.

Larsen, G. K. H., van Foreest, N. D., and Scherpen, J. M. A. (2013). Distributed control of the power supply-demand balance. *IEEE Transactions on Smart Grid*, 4(2):828–836.

Lee, H. G. and Lee, R. (1997). A hybrid approach of linear programming and logic modeling for the market core of sealed bid auctions. *Annals of Operations Research*, 75.

Li, D., Yang, Q., Yu, W., An, D., Zhang, Y., and Zhao, W. (2020). Towards differential privacy-based online double auction for smart grid. *IEEE Transactions on Information Forensics and Security*, 15:971–986.

Liu, X., Yu, C., Zhang, Z., Zheng, Z., Rong, Y., Lv, H., Huo, D., Wang, Y., Chen, D., Xu, J., Wu, F., Chen, G., and Zhu, X. (2021). Neural auction: End-to-end learning of auction mechanisms for e-commerce advertising. In *Proceedings of the*

27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pages 3354–3364.

Lochner, K. M. and Wellman, M. P. (2004). Rule-based specification of auction mechanisms. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*, 2:818–825.

Löding, C. (2021). Automata on infinite trees. In Pin, J., editor, *Handbook of Automata Theory*, pages 265–302. European Mathematical Society Publishing House, Zürich, Switzerland.

Lorini, E. (2016). A minimal logic for interactive epistemology. *Synthese*, 193(3):725–755.

Love, N., Genesereth, M., and Hinrichs, T. (2006). General game playing: Game description language specification. Technical Report LG-2006-01, Stanford University, Stanford, CA.

Lubin, B., Juda, A. I., Cavallo, R., Lahaie, S., Shneidman, J., and Parkes, D. C. (2008). Ice: An expressive iterative combinatorial exchange. *Journal of Artificial Intelligence Research*, 33:33–77.

Maruyama, Y. (2021). A reasoning system for fuzzy distributed knowledge representation in multi-agent systems. In *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6, New York. IEEE.

Maubert, B., Mittelmann, M., Murano, A., and Perrussel, L. (2021). Strategic reasoning in automated mechanism design. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR 2021)*, pages 487–496.

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

Meyer, T., Foo, N., Kwok, R., and Zhang, D. (2004). Logical foundations of negotiation: Outcome, concession and adaptation. *Proceedings of the National Conference on Artificial Intelligence*, pages 293–298.

Mishra, D. and Sharma, T. (2018). A simple budget-balanced mechanism. *Social Choice and Welfare*, 50(1):147–170.

Mittelmann, M., Bouveret, S., and Perrussel, L. (2021a). A general framework for the logical representation of combinatorial exchange protocols (extended abstract). In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2021)*, pages 1602–1604. ACM.

Mittelmann, M., Bouveret, S., and Perrussel, L. (2022a). Representing and reasoning about auctions. *Autonomous Agents and Multi-Agent Systems*, 36(1):20.

Mittelmann, M., Herzig, A., and Perrussel, L. (2021b). Epistemic reasoning about rationality and bids in auctions. In *Proceedings of the European Conference on Logics in Artificial Intelligence (JELIA 2021)*, volume 12678, pages 116–130. Springer.

Mittelmann, M., Maubert, B., Murano, A., and Perrussel, L. (2022b). Automated synthesis of mechanisms. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2022) - to appear*.

Mittelmann, M. and Perrussel, L. (2020a). Auction description language (ADL): general framework for representing auction-based markets. In *Proceedings of the European Conference on Artificial Intelligence (ECAI 2020)*, volume 325, pages 825–832. IOS Press.

Mittelmann, M. and Perrussel, L. (2020b). An epistemic logic for reasoning about strategies in general auctions. In *Proceedings of the Workshops of the International Conference on Logic Programming*, volume 2678.

Mittelmann, M. and Perrussel, L. (2020c). Game description logic with integers: A GDL numerical extension. In *Proceedings of the International Symposium on Foundations of Information and Knowledge Systems (FoIKS 2020)*, volume 12012, pages 191–210. Springer.

Mogavero, F., Murano, A., Perelli, G., and Vardi, M. Y. (2014). Reasoning about strategies: On the model-checking problem. *ACM Transactions on Computational Logic (TOCL)*, 15(4):1–47.

Mogavero, F., Murano, A., Perelli, G., and Vardi, M. Y. (2017). Reasoning about strategies: on the satisfiability problem. *Logical Methods in Computer Science*, 13(1).

Montali, M., Calvanese, D., and De Giacomo, G. (2014). Verification of data-aware commitment-based multiagent system. In *Proceedings of the 14th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2014)*, pages 157–164, Richland. International Foundation for Autonomous Agents and Multiagent Systems.

Myerson, R. B. and Satterthwaite, M. A. (1983). Efficient mechanisms for bilateral trading. *Journal of Economic Theory*, 29(2):265–281.

Narasimhan, H., Agarwal, S. B., and Parkes, D. C. (2016). Automated mechanism design without money via machine learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2016)*.

Nisan, N. (2000). Bidding and allocation in combinatorial auctions. In *ACM Conference on Electronic Commerce*, pages 1–12.

Nisan, N. (2004). Bidding languages. *Combinatorial Auctions*, pages 1–19.

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

Niu, J., Cai, K., Parsons, S., Fasli, M., and Yao, X. (2012). A grey-box approach to automated mechanism design. *Electronic Commerce Research and Applications*, 11(1):24–35.

Novák, P. and Jamroga, W. (2009). Code patterns for agent oriented programming. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2009)*, pages 105–112, Richland. International Foundation for Autonomous Agents and Multiagent Systems.

Okada, N., Todo, T., and Yokoo, M. (2019). Sat-based automated mechanism design for false-name-proof facility location. In *Proceedings of the International Conference on Principles and Practice of Multi-Agent Systems (PRIMA 2019)*.

Parkes, D. C. (2006). Iterative Combinatorial Auctions. In *Combinatorial Auctions*. MIT Press.

Parkes, D. C., Cavallo, R., Elprin, N., Juda, A., Lahaie, S., Lubin, B., Michael, L., Shneidman, J., and Sultan, H. (2005). ICE: An Iterative Combinatorial Exchange. In *Proceedings of the 6th ACM Conference on Electronic Commerce (EC 2005)*, pages 249–258, New York. Association for Computing Machinery.

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

Parsons, S., Rodriguez-Aguilar, J. A., and Klein, M. (2011). Auctions and bidding: A guide for computer scientists. *ACM Computing Surveys*, 43(2).

Pauly, M. (2002). A modal logic for coalitional power in games. *Journal of logic and computation*, 12(1):149–166.

Pauly, M. and Parikh, R. (2003). Game logic-an overview. *Studia Logica*, 75(2):165–182.

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

Phelps, S., McBurney, P., and Parsons, S. (2010). Evolutionary mechanism design: a review. *Autonomous agents and multi-agent systems*, 21(2):237–264.

Pin, J.-É. (2021). *Handbook of Automata Theory*. European Mathematical Society Publishing House, Zuerich.

Pnueli, A. and Rosner, R. (1989). On the Synthesis of a Reactive Module. In *Symposium on the Principles of Programming Languages (POPL 1989)*, pages 179–190, New York. ACM.

Porello, D. and Endriss, U. (2010). Modelling combinatorial auctions in linear logic. In *Proceedings of the Twelfth International Conference on Principles of Knowledge Representation and Reasoning (KR 2010)*, page 71–78. AAAI Press.

Rabe, F. (2006). First-order logic with dependent types. In *Proceedings of the International Joint Conference on Automated Reasoning*, pages 377–391. Springer.

Ramanujam, R. and Simon, S. (2008). Dynamic logic on games with structured strategies. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR 2008)*, page 49–58. AAAI Press.

Reif, J. H. (1984). The complexity of two-player games of incomplete information. *Journal of Computer and System Sciences*, 29(2):274–301.

Rolli, D., Luckner, S., Gimpel, H., and Weinhardt, C. (2006). A Descriptive Auction Language. *Electronic Markets*, 16(1):51–62.

Ruan, J., Van Der Hoek, W., and Wooldridge, M. (2009). Verification of games in the game description language. *Journal of Logic and Computation*, 19(6):1127–1156.

Rubio-Domingo, G. and Linares, P. (2021). The future investment costs of offshore wind: An estimation based on auction results. *Renewable and Sustainable Energy Reviews*, 148:111324.

Saffidine, A. (2014). The game description language is turing complete. *IEEE Transactions on Computational Intelligence and AI in Games*, 6(4):320–324.

Sandholm, T. (2003). Automated mechanism design: A new application area for search algorithms. In *Principles and Practice of Constraint Programming – CP 2003*, pages 19–36, Berlin, Heidelberg. Springer Berlin Heidelberg.

Schobbens, P.-Y. (2004). Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science*, 85(2):82–93.

Schreiber, I. and Romero, B. (2021). *Game Balance*. CRC Press.

Shen, W., Tang, P., and Zuo, S. (2019). Automated mechanism design via neural networks. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2019)*.

Thielscher, M. (2010). A general game description language for incomplete information games. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2010)*, pages 994–999.

Thielscher, M. (2011). GDL-II. *KI - Künstliche Intelligenz*, 25(1):63–66.

Thielscher, M. (2017). GDL-III: A description language for epistemic general game playing. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2017)*, pages 1276–1282.

Thielscher, M. and Zhang, D. (2010). *From General Game Descriptions to a Market Specification Language for General Trading Agents*, pages 259–274. Springer Berlin Heidelberg.

Troquard, N., van der Hoek, W., and Wooldridge, M. (2011). Reasoning about Social Choice Functions. *Journal of Philosophical Logic*, 40(4):473—498.

Troquard, N. and Walther, D. (2012). On satisfiability in ATL with strategy contexts. In *Proceedings of the European Conference on Logics in Artificial Intelligence (JELIA 2012)*.

Van Benthem, J. (2001). Games in dynamic-epistemic logic. *Bulletin of Economic Research*, 53(4):219–248.

van der Hoek, W., Jamroga, W., and Wooldridge, M. (2005). A logic for strategic reasoning. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2005)*, pages 157–164, New York. Association for Computing Machinery.

Vardi, M. Y. (1996). An automata-theoretic approach to linear temporal logic. *Logics for concurrency*, pages 238–266.

Varian, H. R. (2007). Position auctions. *International Journal of industrial Organization*, 25(6):1163–1178.

Vester, S. (2013). Alternating-time temporal logic with finite-memory strategies. In *Proceedings of International Symposium on Games, Automata, Logics, and Formal Verification (GandALF 2013)*, EPTCS, pages 194–207, Borca di Cadore. Open Publishing Association.

Voorneveld, M. (2003). Characterization of pareto dominance. *Operations Research Letters*, 31(1):7 – 11.

Vorobeychik, Y., Reeves, D. M., and Wellman, M. P. (2007). Constrained automated mechanism design for infinite games of incomplete information. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI 2007)*.

Walther, D., van der Hoek, W., and Wooldridge, M. (2007). Alternating-time temporal logic with explicit strategies. In *Proceedings of the Conference on Theoretical Aspects of Rationality and Knowledge (TARK XI)*, pages 269–278, New York. Association for Computing Machinery.

Wang, H., Tang, Y., Liu, J., and Chen, W. (2018). A search optimization method for rule learning in board games. In *Pacific Rim International Conference on Artificial Intelligence*, pages 174–181. Springer.

Wang, Y. and Dechesne, F. (2009). On expressive power and class invariance. *CoRR*, abs/0905.4332.

Wen, C., Xu, M., Zhang, Z., Zheng, Z., Wang, Y., Liu, X., Rong, Y., Xie, D., Tan, X., Yu, C., et al. (2022). A cooperative-competitive multi-agent framework for auto-bidding in online advertising. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1129–1139.

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

Wooldridge, M. and Parsons, S. (2000a). Languages for negotiation. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000)*, page 393–397, NLD. IOS Press.

Wooldridge, M. and Parsons, S. (2000b). On the use of logic in negotiation. In *Proceedings of the Autonomous Agents Workshop on Agent Communication Languages and Conversation Protocols*.

Xia, M., Stallaert, J., and Whinston, A. B. (2005). Solving the combinatorial double auction problem. *European Journal of Operational Research*, 164(1):239–251.

Yadav, N. and Sardiña, S. (2012). Reasoning about agent programs using ATL-like logics. In *Proceedings of the European Conference on Logics in Artificial Intelligence (JELIA 2012)*, pages 437–449, Berlin. Springer Berlin Heidelberg.

Yuan, Y., Wang, F.-Y., and Zeng, D. (2017). Competitive analysis of bidding behavior on sponsored search advertising markets. *IEEE Transactions on Computational Social Systems*, 4(3):179–190.

Zhang, D. (2018). A logic for reasoning about game descriptions. In *AI 2018: Advances in Artificial Intelligence*, pages 38–50, Cham. Springer International Publishing.

Zhang, D. (2020). Behavioural equivalence of game descriptions. In *Australasian Joint Conference on Artificial Intelligence*, pages 307–319. Springer.

Zhang, D. and Thielscher, M. (2015a). A logic for reasoning about game strategies. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI 2015), January 25-30, 2015, Austin, Texas*, pages 1671–1677. AAAI Press.

Zhang, D. and Thielscher, M. (2015b). Representing and reasoning about game strategies. *Journal of Philosophical Logic*, 44(2):203–236.

Zhou, Y., Chakrabarty, D., and Lukose, R. (2008). Budget constrained bidding in keyword auctions and online knapsack problems. In *Internet and Network Economics*, pages 566–576, Berlin, Heidelberg. Springer Berlin Heidelberg.

Zhou, Y. and Lukose, R. (2007). Vindictive bidding in keyword auctions. *ACM International Conference Proceeding Series*, 258:141–146.