A General Framework for the Logical Representation of Combinatorial Exchange Protocols

Munyque Mittelmann  
Université de Toulouse - IRIT  
Toulouse, France  
munyque.mittelmann@irit.fr

Sylvain Bouveret  
Université Grenoble Alpes - LIG  
Grenoble, France  
sylvain.bouveret@imag.fr

Laurent Perrussel  
Université de Toulouse - IRIT  
Toulouse, France  
laurent.perrussel@irit.fr

ABSTRACT
The goal of this paper is to propose a framework for representing and reasoning about the rules governing a combinatorial exchange. Such a framework is of first interest as long as we want to build up digital marketplaces based on auction, a widely used mechanism for automated transactions. Combinatorial exchange is the most general case of auctions, mixing the double and combinatorial variants: agents bid to trade bundles of goods. Hence the framework should fulfill two requirements: (i) it should enable bidders to express their bids on combinations of goods and (ii) it should allow describing the rules governing some market, namely the legal bids, the allocation and payment rules. To do so, we define a logical language in the spirit of the Game Description Language: the Combinatorial Exchange Description Language is the first language for describing combinatorial exchange in a logical framework. The contribution is two-fold: first, we illustrate the general dimension by representing different kinds of protocols, and second, we show how to reason about auction properties in this machine-processable language.

KEYWORDS
Logics for Multi-agents, Game Description Language, Auction-based Markets

1 INTRODUCTION
Auction-based markets are widely used for automated business transactions. There are numerous variants whether we consider single or multiple goods, single or multiple units, single or double-sided [8, 9]. For a fixed set of parameters, the auction protocol, i.e., the bidding, allocation, and payment 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. To do so, auctioneers should at first describe the rules governing an auction and second allow bidders to express complex bids. The aim of this paper is to propose such language with clear semantics enabling us to derive properties. Hereafter, we consider combinatorial exchanges which are the most general case for auctions, mixing double and combinatorial variants [11].

More precisely, such Combinatorial Exchange Description Language should address the following dimensions:

Agent (i) one seller and multiple buyers, or vice-versa (single-side auctions); (ii) multiple sellers and buyers (double-side auctions); (iii) multiple bidders that can be both sellers and buyers (i.e., agents are traders);

Unit (i) single-unit or (ii) multi-unit auction;

Good type (i) single-good or (ii) multiple-goods; for that latter case a bid can consider units and operators such as "1 table and 4 chairs".

Bidding protocol the auction protocol may be (i) one-shot (e.g., sealed-bid auction) or (ii) iterative (e.g., ascending and descending auctions);

Allocation and payment the protocol should (i) detail how the winners are determined [22], (ii) quantify the money transfer (e.g., first price or Vickrey–Clarke–Groves payment [16]).

In the spirit of the General Game Playing [5] where games are described with the help of a logical language, namely the Game Description Language (GDL), we propose the Combinatorial Exchange Description Language (CEDL) which is based on the Auction Description Language (ADL) [13]. CEDL includes a bidding language that can represent a wide range of auctions from a single-side, single-unit and good auction (as a single-unit Vickrey Auction) to Iterative Combinatorial Exchange [16]. As for GDL and ADL, we propose a precise semantics based on state-transition models, that gives a clear meaning to the properties describing an auction. CEDL embeds the Tree-Based Bidding Language [16], which generalizes known languages such as XOR/OR [14] to combinatorial exchange, where agents should be able to express preferences for both buying and selling goods. To the best of our knowledge, CEDL is the first framework offering a unified perspective on an auction mechanism: (i) representation on how to bid and (ii) representation of the protocol including allocation and payment. Such a framework 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.

CEDL extends ADL in numerous ways: ADL only focuses on one-side auctions and cannot represent double auctions; next, ADL is also unable to consider the multiple-good dimension. These two dimensions are now considered in CEDL and thus allow (i) to handle the problem of re-allocating goods, as in an exchange, and (ii) to embed a bidding language to address combinatorial markets.

The paper is organized as follows: in Section 2 we review the key components for describing auctions; we next detail in Section 3 the bidding language that will be embedded in CEDL and exhibit its key properties. Section 4 details CEDL: semantics and syntax. Section 5 specifies two auction protocols with the help of CEDL: a one-shot combinatorial exchange and a variant of ascending combinatorial auction. We conclude by discussing related work and future work for going further.
2 PRELIMINARIES

To describe a combinatorial exchange, we first define an auction signature, that specifies the auction participants (the agents), the goods involved in the auction and the propositions and variables describing each state of the auction:

**Definition 1.** An auction signature \( S \) is a tuple \((N, G, \mathcal{A}, \Phi, Y, I)\), where: (i) \( N = \{1, 2, \ldots, n\} \) is a nonempty finite set of agents (or bidders); (ii) \( G = \{1, \ldots, m\} \) is a nonempty finite set of goods types; (iii) \( \mathcal{A} \) is a nonempty finite set of actions or bid-trees; (iv) \( \Phi = \{P, Q, \ldots\} \) is a finite set of atomic propositions specifying individual features of a state; (v) \( Y = \{y_1, y_2, \ldots\} \) is a finite set of numerical variables specifying numerical features in a state; (vi) \( I = \{\mathbf{z} : \mathbf{z}_\text{min} \leq \mathbf{z} \leq \mathbf{z}_\text{max} \text{ and } \mathbf{z} \in \mathbb{Z}\} \) is a finite subset of integer numbers, denoting the value range of any countable component of the framework, for some arbitrary bounds \( \mathbf{z}_\text{min} \leq \mathbf{z}_\text{max} \).

We will fix an auction signature \( S \) and all concepts will be based on this signature, except if stated otherwise. Note that \( \mathbf{z}_\text{min} \) and \( \mathbf{z}_\text{max} \) in the definition of \( I \), should be large enough to represent the total supply of goods being traded, i.e., as we shall see below, \( \sum_{i \in N} \sum_{j \in G} x_{i,j} \) as well as the cumulative available money among agents. Through the rest of this paper, we assume that this is the case.1

A joint allocation is a tuple \( X = (x_1, \ldots, x_q) \), where \( x_i = (x_{i,1}, \ldots, x_{i,m}) \) is an individual allocation for agent \( i \in N \) and \( x_{i,j} \in I \) denotes the number of units \( j \in G \) held by \( i \). A joint trade is a tuple \( \Lambda = (\lambda_1, \ldots, \lambda_m) \), where \( \lambda_j = (\lambda_{1,j}, \ldots, \lambda_{I,j}) \) is an individual allocation for agent \( i \in N \) and \( \lambda_{i,j} \in I \) denotes the number of units \( j \in G \) being traded by agent \( i \). A trade can be seen as a change over an agent’s initial allocation, resulting in a new one. A positive trade expresses how many units of a good type were purchased and a negative trade represents how many units were sold.

Given an auction signature, we now define a bidding language allowing agents to express the combination of goods they are willing to buy (or sell) and the value they intend to pay (or receive).

3 TREE-BASED BIDDING LANGUAGE

The Tree-Based Bidding Language (TBBL) [11, 16] 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 designed for Combinatorial Exchange. It allows to represent buyers and sellers demands in the same structure. We adopt TBBL as it is a language such as the OR language [2].

Hereafter, we introduce some extra notations to characterize solutions and winners. Let \( T_i \in L_{\text{TBBL}} \) be a bid-tree from bidder \( i \), the set \( \text{Node}(T_i) \) denotes all nodes in the tree, that is, all its inner bids, including \( T_i \) itself. Formally, if \( T_i \in \text{Node}(q, j, o) \), \( \text{Node}(T_i) = \{T_i\} \). Otherwise, \( T_i \) is in the form \( \text{IC}_{\mathcal{I}}(\beta, v') \) and \( \text{Node}(T_i) = \{T_i\} \cup \text{Node}(\beta_1) \cup \cdots \cup \text{Node}(\beta_s) \), where \( s = |\beta| \).

Let \( \beta \in \text{Node}(T_i) \), the set \( \text{Child}(\beta) \subseteq \text{Node}(T_i) \) denotes the children of node \( \beta \). If \( \beta \) is in the form \( \text{IC}_{\mathcal{I}}(\beta, v') \), then \( \text{Child}(\beta) = \{\beta_1, \ldots, \beta_s\} \), where \( s = |\beta| \). Otherwise, \( \text{Child}(\beta) = \{\beta\} \). The leaves of a bid-tree \( T_i \) are denoted by \( \text{Leaf}(T_i) = \{(q, j, o) \in \text{Node}(T_i) : q \in I, o \in I \} \). The value specified at node \( \beta \) is denoted by \( \alpha(\beta) = I \). If \( \beta \) is in the form \( \text{IC}_{\mathcal{I}}(\beta, v') \), then \( \alpha(\beta) = o \). Otherwise, \( \beta \) is in the form \( \text{IC}_{\mathcal{I}}(\beta, v') \) and \( \alpha(\beta) = v' \). Finally, the quantity of units of the good \( j \) specified at a leaf \( \beta = (q, j, o) \) is denoted \( q(\beta, j) = q \). For any other \( \beta \in \text{Leaf}(T_i) \) and \( j \in G \), \( q(\beta, j) = 0 \).

If \( \beta \) is not a leaf (i.e., \( \beta \in \text{Node}(T_i) \) \setminus \text{Leaf}(T_i) \)), then it is in the form \( \text{IC}_{\mathcal{I}}(\beta, v') \) and we denote by \( s(\beta) \) and \( y(\beta) \) the interval-choose constraints \( x \) and \( y \), respectively.

3.1 Trade value and valid solutions

Given a tree \( T_i \) from agent \( i \), the value of a trade \( \lambda_i \in \mathcal{I}_m \) 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 \( \text{sat}_i(\beta) \in \{0, 1\} \) denote whether a node \( \beta \in \text{Node}(T_i) \) is satisfied.
and $s_{ati} = \{ \beta : sat_i(\beta) = 1, \text{ for all } \beta \in \text{Node}(T_i) \}$ denote the nodes satisfied in a solution.

A solution $s_{ati}$ is valid for a tree $T_i$ and trade $\lambda_i$, written $s_{ati} \in valid(T_i, \lambda_i)$ if Rules R1 and R2 hold [11]:

$$x_{\beta}sat_i(\beta) \leq \sum_{\beta'}sat_i(\beta') \leq y_{\beta}sat_i(\beta)$$

$$\forall \beta \in \text{Node}(T_i) \setminus \text{Leaf}(T_i) \quad (R1)$$

$$\sum_{\beta \in \text{Leaf}(T_i)} q_i(\beta, j)sat_i(\beta) \leq \lambda_{i,j}, \forall j \in G \quad (R2)$$

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 a trade $\lambda_i$, given a bid-tree $T_i$, is defined as the solution to the following problem:

$$v_i(T_i, \lambda_i) = \arg\max_{s_{ati}} \sum_{\beta \in \text{Node}(T_i)} v_i(\beta)sat_i(\beta)$$

s.t. R1, R2 hold

3.2 Winner Determination

Given an auction signature, the bid-trees $T_i = (T_{i1}, \ldots, T_{in})$ and a joint allocation $X = (x_{1i}, \ldots, x_{ni})$, where $T_i \in \mathcal{LTBBL}$ denotes the bid-tree from agent $i \in N$, $x_{i} = (x_{i1}, \ldots, x_{in})$ is the individual allocation for $i$ and $x_{i,j} \in I_{j}$ for each good $j \in G$.

The Winner Determination (WD) defines a pair $(\Lambda, sat)$ obtained by the solution to the following mixed-integer program [11]:

$$WD(T, X) : \arg\max_{\Lambda, sat} \sum_{i \in N} \sum_{\beta \in \text{Node}(T_i)} v_i(\beta)sat_i(\beta)$$

s.t. $\lambda_{i,j} + x_{i,j} \geq 0, \forall i \in N, j \in G$ (C1)

$$\sum_{i \in N} \lambda_{i,j} = 0, \forall j \in G$$ (C2)

$$sat_i \in valid(T_i, \lambda_i), \forall i \in N$$ (C3)

$$sat_i(\beta) \in \{0, 1\}, \lambda_{i,j} \in I$$ (C4)

where $sat = (sat_1, \ldots, sat_n)$. Constraint C1 ensures that the joint trade $\Lambda$ is feasible given $X$, that is, no agent sells more items than they initially hold. Constraint C2 imposes strict balance in the supply and demand of goods. Constraint C3 ensures that each individual trade for an agent $i$ is valid given her bid-tree. Constraint C4 defines the range for trades and node satisfaction. We denote by $\Lambda(T, X)$ the joint trade $\Lambda$ in the solution $WD(T, X) = (\Lambda, sat)$, where $\lambda_i(T, X)$ denotes the individual trade for agent $i$ and $\lambda_{i,j}(T, X)$ denotes the units of good $j$ traded by agent $i$.

If there are two or more solutions for $WD(T, X)$, the trade $\Lambda(T, X)$ will be chosen w.r.t. some total order among the elements of $I_{mn}$. This tie-breaking order is omitted to avoid overloading the notation. In the examples, we assume this order is defined such as it is compatible with the Pareto dominance relation [21].

Bid-tree Equivalence. To ensure the finiteness of the framework, we define a maximal subset of TBBL, such that there is no pair of bid-trees representing the exact same value for every trade.

Definition 3. Given the bid-trees $T_i, T'_i \in \mathcal{LTBBL}$ from agent $i \in N$, we say $T_i$ and $T'_i$ are equivalent bid-trees, denoted $T_i \approx tree T'_i$, if for all $\lambda_i \in I_{mn}$, $v_i(T_i, \lambda_i) = v_i(T'_i, \lambda_i)$.

Notice that $\approx_{tree}$ is reflexive, symmetric and transitive.

Proposition 1. Given $\mathcal{LTBBL} \subseteq \mathcal{LTBBL}$, if for all bid-trees $\beta, \beta' \in \mathcal{LTBBL}$ such that $\beta \neq \beta'$, $\beta \not\approx tree \beta'$ then $\mathcal{LTBBL}$ is finite.

Proof. Let $i \in N$ be an agent and $s = |I|$ denote the size of $I$.

Let $T'_i \subseteq \mathcal{LTBBL}$ be such that for all $\beta_i, \beta'_i \in \mathcal{LTBBL}$, where $\beta_i \neq \beta'_i$, we have $\beta_i \not\approx tree \beta'_i$. That is, $v_i(\beta_i, \lambda_i) \neq v_i(\beta'_i, \lambda_i)$ for some $\lambda_i \in I_{mn}$.

For each trade $\lambda_i \in I_{mn}$, there are $s$ possibilities of distinct values $v_i(\beta_i, \lambda_i)$. Furthermore, there are $s^m$ distinct trades. Thus, there may be at most $s^m$ non equivalent bids-tree in $\mathcal{LTBBL}$ and $\mathcal{LTBBL}$ is finite.

Let $\mathcal{LTBBL}_f$ be a maximal subset of $\mathcal{LTBBL}$ such that $\beta \not\approx tree \beta'$, for all $\beta, \beta' \in \mathcal{LTBBL}$, where $\beta \neq \beta'$.

Corollary 1. $\mathcal{LTBBL}_f$ is finite and $|\mathcal{LTBBL}_f| = |I||I_{mn}|$.

A maximal subset of TBBL without equivalent bids is finite, but its size grows exponentially over the size of $I$ and the quantity $m$ of goods in the auction.

4 COMBINATORIAL EXCHANGE DESCRIPTION LANGUAGE

The Combinatorial Exchange Description Language (CEDL) is a framework for specification of auction-based markets and it is composed by a State-Transition Model and a logical language. Next, we present the model, the common legality constraints and the language’s syntax and semantics.

4.1 State-Transition Model

Given an auction signature and the bidding language, we define the auction protocol through a state-transition model. It allows us to represent the key aspects of an auction, at first the legal bids and how to update the auction state.

Let $A \subseteq \mathcal{LTBBL}$ be a finite set of actions. It is not a limitation to assume $A$ finite since a maximal subset of TBBL without equivalent bid-trees is finite. We denote $noop = def (0, 0, 0)$ as the action of not bidding for any good, for some arbitrary $g \in M$.

Definition 4. A state-transition $ST$-model $M$ is a tuple $(W, w, T, L, U, \pi_0, \pi_f)$, where: (i) $W$ is a nonempty set of states; (ii) $w \in W$ is the initial state; (iii) $T \subseteq W$ is a set of terminal states; (iv) $L \subseteq W \times N \times A$ is a legality relation, describing the legal actions at each state, let $L(w, i) = \{ a \in A \mid (w, i, a) \in L \}$ be the set of all legal actions for agent $i$ at state $w$; (v) $U : W \times A \rightarrow W$ is an update function, given $a \in A$, let $d(a)$ be the individual action for agent $i$ in the joint action $d_i(a)$; (vi) $\pi_0 : W \rightarrow 2^A$ is the valuation function for the state propositions; (vii) $\pi_f : W \rightarrow \mathbb{Z}$, is the valuation function for the numerical variables.

Definition 5. Given an $ST$-model $M = (W, w, T, L, U, \pi_0, \pi_f)$, a path is a sequence of states and joint actions $w \rightarrow w_1 \rightarrow \ldots \rightarrow w_n$, such that for any $t \geq 1$: (i) $w_t = \pi_i(w_{t-1});$ (ii) $w_t \neq w_0;$ (iii) $d_i(t) \in L(w_{t-1})$ for any $i \in N$; (iv) $w_t = U(w_{t-1}, d_t);$ and (v) if $w_{t-1} \in T$, then $w_1 = w_t$. 


4.2 Language Syntax

Each payment, allocation and trade should be represented as a numerical variable $y$ in $Y$. We assume the predefined variables set \{$\text{payment}_i, \text{alloc}_i \} : i \in N, j \in G \subseteq Y$. Let $z \in L_\beta$ be a numerical term defined as follows:

\[
z := z' | \text{add}(z, z) | \text{sub}(z, z) | \text{min}(z, z) | \text{max}(z, z) | \text{times}(z, z) | y | \text{win}_{i,j}(\bar{\beta}, \bar{Z}) | \text{value}_i(\beta) | \text{value}_i(\beta, \tilde{z}) | \text{qtd}_i(\beta, j)
\]

where $z' \in I, y \in Y, i \in N, j \in G, \beta \in \mathcal{A}, \bar{\beta} \in \mathcal{A}^0, \bar{Z} \in L_\mathcal{Z}$, and $\mathcal{Z} \in L_{\mathcal{Z}}$.

The terms $\text{add}(z_1, z_2)$, $\text{sub}(z_1, z_2)$, $\text{times}(z_1, z_2)$, $\text{min}(z_1, z_2)$ and $\text{max}(z_1, z_2)$ specify the corresponding mathematical operation or function. For agent $i$ and good $j$, the value of a bid $\beta$, the value of $\beta$ given a trade $\tilde{z}$, the quantity of $j$ in bid $\beta$ and the trade $\lambda_{i,j}(T, X)$ are denoted $\text{value}_i(\beta)$, $\text{value}_i(\beta, \tilde{z})$, $\text{qtd}_i(\beta, j)$ and $\lambda_{i,j}(T, X)$, respectively.

The Combinatorial Exchange Description language is denoted by $\mathcal{L}_{CEDL}$ and a formula $\varphi$ in $\mathcal{L}_{CEDL}$ is defined by the following BNF grammar:

\[
\varphi ::= p | \text{initial} | \text{terminal} | \text{legal}_i(\beta) | \text{does}_i(\beta) \\
  \neg \varphi | \varphi \land \varphi | \top | \| \varphi | \bot | \varphi \lor \varphi
\]

where $p \in \Phi, i \in N, res \in \{\text{buyer, seller, good, unit}\}, \beta \in \mathcal{A}$ and $\beta \in L_\mathcal{Z}$.

Intuitively, initial and terminal specify the initial terminal states, resp.: $\text{legal}_i(\beta)$ asserts that agent $i$ is allowed to take action $\beta$ at the current state and $\text{does}_i(\beta)$ asserts that agent $i$ takes action $\beta$ at the current state. The formula $\Box \varphi$ means “$\varphi$ holds at the next state”. The formulas $z_1 > z_2$, $z_1 < z_2$, $z_1 = z_2$ mean that a numerical term $z_1$ is greater, less and equal to a numerical term $z_2$, respectively. The formula $\text{rest}_i(\text{res}_i, \beta)$ specifies whether the bid $\beta$ from agent $i$ respects the restriction $\text{res}_i \in \{\text{buyer, seller, good, unit}\}$. The restriction $\text{buyer}$ specifies that $\beta$ cannot have negative quantities or prices. Similarly, the restriction $\text{seller}$ specifies that $\beta$ cannot have positive quantities or prices. The restriction $\text{good}$ states that $\beta$ should be a leaf node. Finally, the restriction $\text{unit}$ says any leaf node in $\beta$ can only demand a single unit from a good type.

Other connectives $\lor, \rightarrow, \leftrightarrow, \top, \bot$ are defined by $\neg$ and $\land$ in the standard way.

4.3 Language Semantics

The semantics for the CEDL language is given in two steps. First, we define Function $f$ to compute the meaning of numerical terms $z \in L_\mathcal{Z}$ in some specific state. Next, a formula $\varphi \in \mathcal{L}_{CEDL}$ is interpreted with respect to a step in a path.

Definition 6. Given an ST-model $M$, define Function $f : \mathcal{L}_\mathcal{Z} \times W \rightarrow \mathcal{Z}$, assigning any $z \in L_\mathcal{Z}$, and state $w \in W$ to a number in $\mathcal{Z}$.

If $z$ is in the form $\text{add}(z', z'')$, $\text{sub}(z', z'')$, $\text{min}(z', z'')$, $\text{max}(z', z'')$, or $\text{times}(z', z'')$, then $f(z, w)$ is defined through the application of the corresponding mathematical operators and functions over $f(z', w)$ and $f(z'', w)$. Otherwise, $f(z, w)$ is defined as follows:

\[
f(z, w) = \begin{cases} 
  v_i(T_i) & \text{if } z = \text{value}_i(T_i) \\
  q_i(T_i, \tilde{z}) & \text{if } z = \text{qtd}_i(T_i, j) \\
  z & \text{if } z = \text{win}_{i,j}(T, X)
\end{cases}
\]

Definition 7. Let $M$ be an ST-Model. Given a path $\delta$ of $M$, a stage $t$ on $\delta$ and a formula $\varphi \in \mathcal{L}_{CEDL}$, we say $\varphi$ is true (or satisfied) at $t$ under $\delta$ under $M$, denoted by $M, \delta, t \models \varphi$, according to the following definition:

\[
M, \delta, t \models p \iff p \in \pi_0(\delta[t]) \\
M, \delta, t \models \neg \varphi \iff \neg (M, \delta, t \models \varphi) \\
M, \delta, t \models \text{legal}_i(\beta) \iff M, \delta, t \models \text{does}_i(\beta) \\
M, \delta, t \models \text{initial} \iff \delta[t] = \top \\
M, \delta, t \models \text{terminal} \iff \delta[t] = \bot \\
M, \delta, t \models \text{legal}_i(\alpha) \iff a \in \mathcal{L}(\delta[t], i) \\
M, \delta, t \models \text{does}_i(\alpha) \iff \theta_i(\delta, t) = a \\
M, \delta, t \models \Box \varphi \iff \exists \delta', t + 1 \models \varphi \\
M, \delta, t \models \text{rest}_i(\text{buyer}, a) \iff \forall \beta \in \text{Node}(a), v_i(\beta) \geq 0 \\
M, \delta, t \models \text{rest}_i(\text{seller}, a) \iff \forall \beta \in \text{Node}(a), v_i(\beta) \leq 0 \\
M, \delta, t \models \text{rest}_i(\text{good}, a) \iff \text{Child}(a) = \emptyset \\
M, \delta, t \models \text{rest}_i(\text{unit}, a) \iff \forall \beta \in \text{Leaf}(a), \exists j \in G, q_i(\beta, j) \in \{0, 1\}
\]

A formula $\varphi$ is globally true through $\delta$, denoted by $M, \delta \models \varphi$, if $M, \delta, t \models \varphi$ for any stage $t$ of $\delta$. A formula $\varphi$ is globally true in an ST-Model $M$, written $M \models \varphi$, if $M, \delta \models \varphi$ for all paths $\delta$ in $M$. Finally, let $\Sigma$ be a set of formulas in $\mathcal{L}_{CEDL}$, then $M$ is a model of $\Sigma$ if $M \models \varphi$ for all $\varphi \in \Sigma$.

The following propositions show that if a player bids at a stage in a path, then (i) she does not bid anything else at the same stage and (ii) the bid is legal. Additionally, the value of the bid-tree noop is zero. Notice that an agent bidding noop does not imply her payment will be zero (e.g., there may be fees for participating).

Proposition 2. For each agent $i \in N$ and each bid-tree $\beta \in \mathcal{A}$,

1. $M \models \text{does}_i(\beta) \rightarrow \neg \text{does}_i(\beta')$ for any $\beta' \in \mathcal{A}$ such that $\beta' \neq \beta$
2. $M \models \text{does}_i(\beta) \rightarrow \text{legal}_i(\beta)$

Proof. For Statement 1, assume $M, \delta, t \models \text{does}_i(\beta)$ iff $\delta_i(\delta, t) = \beta$. Then for any $\beta' \neq \beta \in \mathcal{A}_\mathcal{Z}, \theta_i(\delta, t) = \beta'$ and $M, \delta, t \models \neg \text{does}_i(\beta')$. 

Let us verify Statement 2. Assume \( M, \delta, t \models \text{does}_i(\beta) \), then \( \theta_i(\delta, t) = \beta \) and by the definition of \( \delta, \beta \in L(\delta[r], t) \) and \( M, \delta, t \models \text{legal}_i(\beta) \).

\[ \square \]

**Lemma 1.** For each agent \( i \in N \), each bid-tree \( \beta \in A \) and each \( z \in L^\mathbb{N} \), \( M \models \text{value}_i(\text{noop}) = 0 \wedge \text{value}_i(\text{noop}, z) = 0. \)

**Proof.** We consider Statement 1. 

Let us first show how to represent some classical but important properties from Mechanism Design, namely budget-balance, no-deficit and individual rationality conditions.

**Budget-Balanced Mechanisms.** A mechanism is Budget-Balanced (BB) if the cumulative payment among the bidders is zero, for every valuations they may have [12]. Given an ST-model \( M \), if \( M \) satisfies the no-deficit condition according to the validity of the following formula:

\[
\forall i \in N, \exists \bar{\lambda}, \gamma, \exists \theta, \exists \phi, \forall z \in L^\mathbb{N}, \forall \delta, \forall \beta \in L(\delta), \forall t \in T, \exists \gamma, \exists \phi, \forall z \in L^\mathbb{N}, \forall \delta, \forall \beta \in L(\delta), \forall t \in T, M \models \text{value}_i(\text{noop}) = 0 \wedge \text{value}_i(\text{noop}, z) = 0.
\]

**No-deficit Mechanisms.** A mechanism where only the designer can earn revenue satisfies no-deficit [12]. The no-deficit condition is a relaxation from BB, where the cumulative payment among the bidders cannot be negative. In CEDL, an ST-model \( M \) satisfies the no-deficit condition according to the validity of the following formula:

\[
\text{noDeficit} \equiv \text{add}(\text{payment}_1, \ldots, \text{payment}_n) = 0.
\]

**Individual Rationality.** A mechanism is individually rational if agents can always achieve at least as much utility as from participating as without participating [17]. To represent such condition, we assume each agent \( i \in N \) has a private valuation in \( N \) for each individual trade \( \lambda_i \in L^\mathbb{N} \), denoted \( \delta_i(\lambda_i) \). As [11], we also assume the agents have monotonic valuations, so that \( \delta_i(\lambda_i') \geq \delta_i(\lambda_i) \), for any trade \( \lambda_i' \geq \lambda_i \) (i.e., \( \delta_i(\lambda_i') \geq \delta_i(\lambda_i) \), for each \( j \)). The agent’s utility is quasi-linear, denoted \( u_i(\lambda_i, p_i) = \delta_i(\lambda_i) - p_i \), where \( p_i \) denotes \( i \)’s payment. 

Rephrased in terms of ST-models, we say a model \( M \) is Individual Rational IR if it is Individual Rational IR for each agent \( i \in N \) in each path \( \delta \) in \( M \) and stages \( t \in \delta \).

A stage \( t \) of \( \delta \) is \( \delta_i \), written \( M, \delta, t \models \delta_i \) if there is a path \( \delta' \) in \( M \) such that \( \delta[0,t] = \delta'[0,t] \), \( \theta_i(\delta, t) = \theta_i(\delta', t) \), for all \( r \in N \setminus \{i\} \), and \( M, \delta', t \models \text{utility}_i(\text{trade}, \text{payment}) = x \rightarrow \text{utility}_i(\text{trade}, \text{payment}) \geq x \), for each \( x \in F \) in the initial state. In other words, \( IR \) reports meta-reasoning as choices among paths have to be considered.

Let us now represent in CEDL two types of auction-based markets: a One-Shot Combinatorial Exchange and a Simultaneous Ascending Auction. For both of them, we detail the rules representation, the semantic representation and we revisit the Mechanism Design conditions.

### 5.1 Representing a Combinatorial Exchange

To represent a One-Shot Combinatorial Exchange with multiple units of \( m \) goods and \( n \) players, we first describe the auction signature, written \( S_{ce} = (N_{ce}, G_{ce}, A_{ce}, \Phi_{ce}, \gamma_{ce}, \iota_{ce}) \), where \( N_{ce} = \{1, \ldots, n\} \), \( G_{ce} = (1, \ldots, m) \), \( A_{ce} \subseteq L_{\text{TBBLL}}f \), \( \Phi_{ce} = \{\text{bidRound}\} \), \( \gamma_{ce} = \{\text{alloc}_i,j, \text{trade}_i,j, \text{payment}_i:j \in N_{ce}, i \in G_{ce}, \} \) and \( \iota_{ce} \in \mathbb{Z} \).

Each instance of a One-Shot Combinatorial Exchange is specific and is defined with respect to \( A_{ce}, \iota_{ce} \) and the constant values \( n, m \in \mathbb{N}_{\iota_{ce}} \} (\text{the size of } \mathbb{N}_{ce} \text{ and } \mathbb{G}_{ce} \text{, resp.}) \), and \( x_{i,j} \in \mathbb{N}_{ce} \), for each \( i \in N_{ce} \) and \( j \in G_{ce} \). Each constant \( x_{i,j} \) represents the quantity of units of \( j \) initially held by agent \( i \). The rules of a One-Shot Combinatorial Exchange are represented by CEDL-formulas as shown in Figure 2.

Figure 2: A Combinatorial Exchange represented by \( \Sigma_{ce} \)

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 \( \text{bidRound} \) 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 \( \text{noop} \) ). Once in a terminal state, players can only do \( \text{noop} \). Otherwise, they can bid any \( \text{bid-tree} \) in \( A_{ce} \) (Rules 3 and 4). If a list of \( \text{bid-trees} \) is the joint action performed in the initial state, then in the next state each agent receives an individual trade, which is assigned by the WD over the initial allocations and the bid-trees (Rule 5). After performing a bid in the initial state, the payment for an agent will be the value of her trade given her bid (Rule 6). No numerical variable has its value changed after reaching a terminal state (Rule 7). The allocation for an agent is the quantity of goods she initially held plus her trade (Rule 8). Finally, the proposition \( \text{bidRound} \) is always false in the next state (Rule 9).

**Representing as a model.** Next, we address the model representation. Let \( M_{ce} \) be the set of ST-models \( M \) defined for any \( A_{ce} \subseteq L_{\text{TBBLL}}f \), \( \iota_{ce} \in \mathbb{Z} \), and the constants \( n, m \in \mathbb{N}_{\iota_{ce}} \} \) and \( x_{i,j} \in \mathbb{N}_{ce} \), for each \( i \in N_{ce} \) and \( j \in G_{ce} \).

Each \( M_{ce} \) is defined as:

- For every \( i \in N_{ce} \) and \( j \in G_{ce} \):
  - \( w_{ce} = \{ (b, x_{1,i}, \ldots, x_{m,i}, l_{1,i}, \ldots, l_{n,i}, p_{1,i}, \ldots, p_{n,i}) : b \in \{0, 1\}, x_{i,j} \in \mathbb{N}_{ce}, p_{i,j} \in \mathbb{N}_{ce} \} \); 
  - \( w_{ce} = \{ (b, x_{1,i}, \ldots, x_{m,i}, l_{1,i}, \ldots, l_{n,i}, p_{1,i}, \ldots, p_{n,i}) : b \in \{0, 1\}, x_{i,j} \in \mathbb{N}_{ce}, p_{i,j} \in \mathbb{N}_{ce} \} \);
  - \( T_{ce} = \{ w : w \in w_{ce} \} \).
\( L_{ce} = \{(w, i, noop) : i \in N_{ce} \land w \in T_{ce}\} \cup \{(\omega_{ce}, i, \beta) : \beta \in A_{ce} \land i \in N_{ce}\} \)

\( U_{ce} \) is defined as follows: for all \( w = (b, x_{1,1}, \ldots, x_{n,m}, \lambda_{1,1}, \ldots, \lambda_{n,m}, p_1, \ldots, p_n) \in W_{ce} \) for all \( d \in A_{ce} \):

- If \( w = \omega_{ce} \), then \( U_{ce}(w, d) = \langle 0, x_{1,1}, \ldots, x_{n,m}, \lambda_{1,1}', \ldots, \lambda_{n,m}', p_1', \ldots, p_n' \rangle \), where \( \lambda_{ij}' = \lambda_{ij} - d(x_{ij+1}, \ldots, x_{n,m}) \) for each \( i \in N_{ce} \) and \( j \in G_{ce} \).

- Otherwise, \( U_{ce}(w, d) = w \).

For each \( w \in W_{ce}, i \in N_{ce} \) and \( j \in G_{ce} \), \( \pi_{Y_{ce}}(w, trade_{ij}) = \lambda_{ij} = \pi_{Y_{ce}}(w, alloc_{ij}) = x_{ij} \) and \( \pi_{Y_{ce}}(w, payment) = p_i \), for each \( i \in N_{ce} \) and \( j \in G_{ce} \).

Hereafter, we assume an instance of \( M_{ce} \in M_{ce} \) and \( \Sigma_{ce} \) for some \( A_{ce} \subseteq \mathcal{LBFL} \). \( L_{ce} \subseteq Z, m, n \in L_{ce} \cup \{0\} \) and \( x_{ij} \in L_{ce} \), where \( i \in N_{ce} \) and \( j \in G_{ce} \).

**Example 1.** Let \( M_{ce} \in M_{ce} \), where \( (i) n = 2 \) and the agents are denoted by \( r_1 \) and \( r_2 \), \( (ii) m = 2 \) and the good types are denoted by \( a \) and \( b \), and \( (iii) x_{1,1} = 0, x_{1,2} = 1, x_{2,1} = 2 \) and \( x_{2,2} = 0 \), i.e., at the beginning of the auction, agent 1 has 1 unit of \( a \) and agent 2 has 2 units of \( a \). Figure 3 illustrates a path in \( M_{ce} \), where the agents perform the bids previously introduced in Figure 1. In state \( w_0 \), all the payments and trades are zero. Their joint bid leads to state \( w_1 \), where the joint trade obtained by the winner determination is \( (2, -1, 2, -1) \). The tie-breaking ensures that the joint trade is unique. Given the joint trade, the allocation for agent 1 is 2 units of \( a \) and the allocation for agent 2 is 1 unit of \( b \). Since \( w_1 \) is terminal, the agents can only bid noop and the state that succeeds \( w_1 \) is \( w_1 \) itself.

**Figure 3: A Path in \( M_{ce} \), with 2 bidders and 2 goods**

Let us now evaluate the protocol. First, Proposition 3 shows that \( M_{ce} \) is a sound representation of \( \Sigma_{ce} \).

**Proposition 3.** \( M_{ce} \) is an ST-model and it is a model of \( \Sigma_{ce} \).

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

- Let us verify Rule 1. Assume \( M_{ce}, \delta, t \models \varphi \), then \( \delta[t] = \omega_{ce} \), i.e., \( \delta[t] = \langle 1, x_{1,1}, \ldots, x_{n,m}, 0, \ldots, 0, \ldots, 0 \rangle \). By the definitions of \( \pi_{Y_{ce}} \) and \( \pi_{\phi_{ce}} \), \( \pi_{\phi_{ce}}(\omega_{ce}) = \{\text{bidRound}, \pi_{Y_{ce}}(\omega_{ce}, \text{payment}) = 0 \land \pi_{Y_{ce}}(\omega_{ce}, \text{trade}_{ij}) = 0 \text{ for all } i \in N_{ce} \text{ and } j \in G_{ce} \} \).

Next we verify Rule 4. Assume \( M_{ce}, \delta, t \models \varphi \), then \( \delta[t] = \omega_{ce} \) and for all \( i \in N_{ce} \) and \( \beta \in A_{ce} \), \( \omega_{ce}, i, \beta \in L_{ce} \). Thus, \( M_{ce}, \delta, t \models \text{bidRound} \land \bigwedge_{i \in N_{ce}} \text{payment}_i = 0 \land \bigwedge_{i \in G_{ce}} \text{trade}_{ij} = 0 \).

Then we consider Rule 5. \( M_{ce}, \delta, t \models \text{does}(T_1, \ldots, T_n) \land \text{initial} \), for \( (T_1, \ldots, T_n) \in A_{ce} \), i.e., \( M_{ce}, \delta, t \models \bigwedge_{i \in N_{ce}} \text{does}(T_i) \) and \( M_{ce}, \delta, t \models \text{initial} \). Thus, \( \delta[t] = \omega_{ce} \), for all \( i \in N_{ce} \). The update function \( U_{ce} \) defines \( \delta[t+1] \) such that \( \pi_{Y_{ce}}(\delta[t+1], \text{trade}_{ij}) = \lambda_{ij}(T_1, \ldots, T_n, x_{1,1}, \ldots, x_{n,m}) \), for each \( i \in N_{ce} \) and \( j \in G_{ce} \). Thus, \( M_{ce}, \delta, t + 1 \models \bigwedge_{i \in N_{ce}} \text{does}(T_i, \ldots, T_n, x_{1,1}, \ldots, x_{n,m}) \) and also \( M_{ce}, \delta, t \models \bigwedge_{i \in N_{ce}} \text{does}(T_i, x_{1,1}, \ldots, x_{n,m}) \).

Finally, we consider Rule 8. Let \( i \in N_{ce} \) and \( j \in G_{ce} \). If \( i = 0 \), then \( \delta[t] = \omega_{ce} \). By the valuation function \( \pi_{Y_{ce}} \), \( \pi_{Y_{ce}}(\omega_{ce}, \text{alloc}_{ij}) = x_{ij} \) and \( \pi_{Y_{ce}}(\omega_{ce}, \text{trade}_{ij}) = 0 \). If \( i = 1 \), then by the path definition we have \( \delta[t] = U_{ce}(\omega_{ce}, d) \), for some \( d \in A_{ce} \). The update function \( U_{ce} \) defines \( \pi_{Y_{ce}}(\omega_{ce}, \text{alloc}_{ij}) = x_{ij} + \text{trade}_{ij} \). Otherwise, for any \( j > 1 \), \( \delta[t] \in T_{ce} \) (see Rule 7 and path definition).

**Proposition 4.** For each agent \( i \in N_{ce} \) and bid-tree \( \beta' \),

1. \( M_{ce} \models \text{initial} \rightarrow \neg \text{terminal} \)
2. \( M_{ce} \models \text{legal}_{i}(\beta) \rightarrow \neg \text{legal}_{i}(\beta') \), for any \( \beta' \in A_{ce} \) such that \( \beta' \neq \text{noop} \)

**Proof.** Given a path \( \delta \) in \( M_{ce} \) and a stage \( t \) of \( \delta \). Let us verify Statement 1. Assume \( M_{ce}, \delta, t \models \text{initial} \). Then \( \delta[t] = \omega_{ce} \). By the path definition, for any \( j \geq 1 \), \( \delta[j] \neq \omega_{ce} \). By the construction of \( T_{ce} \), we have \( T_{ce} = W_{ce} \setminus \{\omega_{ce}\} \). Thus, \( M_{ce}, \delta, t + 1 \models \text{terminal} \) and \( M_{ce}, \delta, t \models \neg \text{terminal} \).

Now we verify Statement 2. Assume \( M_{ce}, \delta, t \models \text{legal}_{i}(\beta) \). From Statement 1 and since the path construction defines a loop in any terminal state (i.e, if \( \delta[t] \in T_{ce} \) then \( \delta[t] = \delta[t+1] \)), we have that \( \delta[t+1] \in T_{ce} \). Thus, \( L_{ce}(\delta[t+1], i) = \{\text{noop} \} \) and \( M_{ce}, \delta, t + 1 \models \neg \text{legal}_{i}(\beta') \), for any \( \beta' \in A_{ce} \) such that \( \beta' \neq \text{noop} \).
We then focus on budget balance, non-deficit and individual rationality conditions.

**Theorem 1.** \( M_{ce} \models \neg \text{BB} \) and \( M_{ce} \models \neg \text{noDeficit} \).

**Proof.** Let \( \delta \) be a path in \( M_{ce} \) and \( t \) be a stage in \( \delta \).

If \( \delta[t] = \bar{w}_{ce} \), then \( M_{ce}, \delta, t \models \bigland_{i \in N_{ce}} \text{payment}_i = 0 \). Thus, \( M_{ce}, \delta, t \models \text{BB} \) and \( \delta[t] = \text{noDeficit} \).

Otherwise, by the path definition, \( \delta[t] = U_{ce}(\delta[t-1], \theta(\delta, t-1)) \).

Since \( M_{ce} \models \neg \text{initial} \rightarrow \text{terminal} \) and given that \( \delta[t-1] = \delta[t] \) whenever \( \delta[t-1] \in T_{ce} \), we focus on the case where \( \delta[t-1] = \bar{w}_{ce} \) and the remaining cases follow by consequence.

Let us denote \( T_i = \emptyset(\delta, t-1) \), for each \( i \in N_{ce} \). By Rules 5 and 6 from \( \Sigma_{ce} \), we have \( M_{ce}, \delta, t \models \bigland_{i \in N_{ce}} (\text{trade}_i = \text{win}_i(T_1, \ldots, T_n, x_1, \ldots, x_{nm})) \wedge \text{payment}_i = \text{value}_i(T_1, \text{trade}_i) \).

The solution is \( W(D(T_1, \ldots, T_n, x_1, \ldots, x_{nm})) = (\Lambda, \text{sat}) \) satisfies Constraints C1-C4 and maximizes \( \sum_{i \in N_{ce}} \sum_{\bar{v}_i(\text{Node}(T_i))} (v_i(\text{sat}_i(T_1, \lambda_1))^\prime) \), which is, maximizes \( \sum_{i \in N_{ce}} (v_i(T_i))^\prime \).

We define the pair \( (\Lambda, \text{sat}) \), such that \( \Lambda^\prime = 0 \), is an empty joint trade and \( \text{sat}^\prime = (\text{sat}_1, \ldots, \text{sat}_n) \), where \( \text{sat}_i' = \{ \} \), for all \( i \in N_{ce} \). Notice \( \text{sat}_i' \) is valid for \( T_i^1 \) and \( \lambda_1^i \) (i.e., \( \text{sat}_i' \in \text{valid}(T_i^1, \lambda_1^i) \)) and \( \sum_{i \in N_{ce}} v_i(\text{sat}_i(T_1, \lambda_1))^\prime = 0 \). Remind that the value of a trade \( \lambda^i \) given a bid-tree \( T_i \) maximizes the value of the satisfied nodes in a valid solution. Thus, \( v_i(T_i, \lambda_i^i) \geq 0 \), (i.e., it is at least equal to \( \sum_{i \in \text{Node}(T_i)} v_i(\text{sat}_i(T_1, \lambda_1))^\prime \)).

Since the pair \( (\Lambda, \text{sat}) \) satisfies the Constraints C1-C4, a solution \((\Lambda, \text{sat})\) for \( W(D(T_1, \ldots, T_n, x_1, \ldots, x_{nm})) \) is unique.

Now, let us assume \( \delta[t] \neq \bar{w}_{sa} \). Let \( x \in \text{lsa} \) such that \( M_{ce}, \delta, t \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \) from the path definition, \( \delta[t] = \delta[t+1] \) and thus \( M_{ce}, \delta, t+1 \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \).

**5.1.1 Vickrey–Clarke–Groves payment.** A Vickrey-Clarke-Groves (VCG) mechanism computes a discount for each winner’s payment, such as she has an incentive to be truthful: the bidder is willing to reveal her private value. Remind that this work focuses on the auction definition and not on the bidder’s behavior, the reader may refer to Krishna (2009) for this strategic aspect.

Let us show how to express VCG payments for an agent \( i \in N_{ce} \). Given the bid-trees \( T = (T_1, \ldots, T_n) \) and the initial allocation \( X \in \mathbb{P}^m \) let \( T_i = (T_1, \ldots, T_n) \) be defined as follows: \( T_i = \text{noop} \) and \( T'_i = T_i \) if \( T_i \in N_{ce} \). Similarly, let \( X' = (x_1', \ldots, x_{nm}') \) be defined as follows: \( x_{ij}' = 0 \) and \( x_{ij}' = x_{ij} \), for all \( i \in N_{ce} \). Remind \( \lambda_i(T, X) \) denotes the individual trade of agent \( i \) in the solution for \( W(D(T, X)) \).

The VCG payment for agent \( i \) is the value of the bid-tree \( T_i \) given the individual trade \( \lambda_i(T, X) \) discounted by the difference between the cumulative values from the joint trade \( \Lambda(T, X) \) and the trade resulting from removing the bid and allocation of \( i \). Formally,

\[
\text{p}_{vcg\,i} = v_i(T_i, \lambda_i(T, X)) - \sum_{r \in N_{ce}} (v_r(T_r, \lambda_r(T', X')) - v_r(T'_r, \lambda_r(T', X'))) \]

To construct a combinatorial exchange with VCG payments, we can define \( \Sigma_{vcg} \) such that it is defined exactly as \( \Sigma_{ce} \), except by Rule 6, which is replaced by the following:

\[
\text{does}(T) \wedge \neg \text{terminal} \wedge p_i = \text{sub(}\text{value}_i(T_i, \text{win}_i(T, X)))
\]

\[
\text{add(}\text{value}(T_1, \text{win}(T, X)), \text{value}(T_i, \text{win}(T, X'), X')) 
\]

\[
\text{sub(}\text{value}(T_n, \text{win}(T, X)), \text{value}(T_n, \text{win}(T, X')))
\]

\[
\rightarrow \bigland_{i \in N_{ce}} (v_i(T_i, \lambda_i(T, X)) - v_i(T'_i, \lambda_i(T', X')))
\]

for each \( p_i \in I_{ce}, T \in \mathbb{P}^m \) and \( i \in N_{ce} \).

Let \( M_{vcg} \) be an ST-model defined as \( M_{ce} \), except by the definition of \( \pi_{vcg}(w, \text{payment}_i) \), for all \( i \), which are defined by according to the VCG price \( p_{vcg\,i} \) in a state \( w \in \mathcal{W}_{vcg} \). Unsurprisingly, \( BB \) and \( \text{noDeficit} \) are not valid in \( M_{vcg} \).

**Proposition 6.** \( M_{vcg} \models \neg \text{BB} \) and \( M_{vcg} \models \neg \text{noDeficit} \).

**Proof.** Let us prove it by showing a counterexample. Assume \( n = 2, m = 1, x_1, x_2 = 0 \) and \( x_1, x_2 = 1 \), i.e., there are two agents, one good type and the second agent initially holds one copy of the good. Let \( T_1 = (1, 1, 2) \) and \( T_2 = (1, 1, 1) \). Let \( \delta \) be a path such that \( M_{vcg}, \delta, t \models \text{initial} \wedge \text{does}(T_1) \wedge \text{does}(T_2) \).

Thus, \( \text{v}_1(T_1, 1) = 2 \) and \( \text{v}_1(T_2, 1) = 1 \). By the update function, we have \( \text{v}_1(T_1, 1) = 1 \wedge \text{trade}_1 = 1 \) and \( \text{v}_1(T_2, 1) = 1 \wedge \text{trade}_1 = 1 \). Note no trade is performed when any of the agents does not participate, i.e., \( \Lambda(T_1, T_2, 0, 1) = \Lambda(T_1, T_2, 0, 0) = 0 \).

Now, let us assume \( \delta[t] \neq \bar{w}_{sa} \). Let \( x \in \text{lsa} \) such that \( M_{ce}, \delta, t \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \) from the path definition, \( \delta[t] = \delta[t+1] \) and thus \( M_{ce}, \delta, t+1 \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \).

Now, let us assume \( \delta[t] \neq \bar{w}_{sa} \). Let \( x \in \text{lsa} \) such that \( M_{ce}, \delta, t \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \) from the path definition, \( \delta[t] = \delta[t+1] \) and thus \( M_{ce}, \delta, t+1 \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \).

Now, let us assume \( \delta[t] \neq \bar{w}_{sa} \). Let \( x \in \text{lsa} \) such that \( M_{ce}, \delta, t \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \) from the path definition, \( \delta[t] = \delta[t+1] \) and thus \( M_{ce}, \delta, t+1 \models \text{utility}_i(\text{trade}_i, \text{payment}_i) = x \).
5.2 Representing a Simultaneous Ascending Auction

Let us now consider a new type of auction: the Simultaneous Ascending Auction (SAA) is a single-side and single-unit auction similar to the traditional English Auction, except that several goods are sold at the same time, and that the bidders simultaneously bid for any number of goods they want [4]. To represent an SAA with m good types and n agents, we first describe the auction signature, written $\Sigma_{sa} = (N_{sa}, G_{sa}, A_{sa}, \Phi_{sa}, Y_{sa}, lsa)$, where $N_{sa} = \{1, \ldots, n\}$, $G_{sa} = \{1, \ldots, m\}$, $A_{sa} \subseteq L_{TBBLf}$, $\Phi_{sa} = \{\text{sold}_j, \text{bid}_{i,j} : j \in G_{sa}$ and $i \in N_{sa}\}$ and $Y_{sa} = \{\text{price}, \text{price}_j, \text{trade}_{i,j}, \text{alloc}_j, \text{payment}_i \}$. The propositions sold$_j$ and bid$_{i,j}$ represent whether the good $j$ was sold and whether $i$ is bidding for $j$, respectively. The variables price and price$_j$ specify the current price for any unsold good and the selling price for $j$, respectively.

Each instance of an SAA is specific and defined with respect to $\Sigma_{sa}$ and the values $inc, n, m \in lsa + \{0\}$ and start in $lsa$, representing the quantity of agents and goods, the increment, and the starting price, respectively. Let max$_{sa}$ denote the largest value in $lsa$. Then, the rules of an SAA are formulated by CEDL-formulas as shown in Figure 4.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{example.png}
\caption{Simultaneous Ascending Auction represented by $\Sigma_{sa}$}
\end{figure}

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 there is a trade for 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 if she is the only one bidding for it, otherwise there is no trade (Rules 4-5). For each good, an agent can either bid the value 0, 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, (iii) there is no allocation, and (iv) the active bidders for each good are updated w.r.t. their bids (Rules 7-10). The allocations are assigned in terminal states w.r.t. trades (Rule 11). The payment for an agent is the cumulative value of the selling price for her traded goods (Rule 12). Finally, after a terminal state, a numerical variable cannot change (Rule 13). Let $\Sigma_{sa}$ be the set of Rules 1-13.

\textbf{Representing a model.} Next, we address the model representation of the Simultaneous Ascending Auction (SAA). Let $\Sigma_{sa}$ be the set of ST-models $M_{sa}$ defined for any $\Sigma_{sa} \subseteq L_{TBBLf}$, $lsa \subset \emptyset$, and the constant values inc, start in $lsa +$ and $n, m, n \in lsa + \{0\}$. Let max$_{sa}$ denote the largest value in $lsa$. Each $M_{sa}$ is defined as follows:

- $W_{sa} = \{(b_{1,1}, \ldots, b_{m,n}, t_{1,1}, \ldots, t_{m,n}, p, p_1, \ldots, p_m) : \text{\textbf{bid} } i, j \in \{0, 1\} \cup \{p, p_j \in lsa, & i \in Nsa, \} \in Gsa\}$
- $\omega_{sa} = \{(0, 0, 0, 0, 0, \text{start}, \ldots, \text{start})\}$
- $T_{sa} = \{w : w = (b_{1,1}, \ldots, b_{m,n}, t_{1,1}, \ldots, t_{m,n}, p, p_1, \ldots, p_m) \in W_{sa} \\cup \{\text{\textbf{start} } i, j \in \{0, 1\} \cup \{p, p_j \in lsa, & i \in Nsa, \} \in Gsa\}$, either (i) $t_{i,j} = 1$ for some $i \in Nsa$ or (ii) bid$_{i,j} = 0$, for all $i \in Nsa$
- $L_{sa} = \{(w, i, OR((\langle 1, 1, p_1 \rangle, \ldots, (1, 1, p_m)), 0): i \in Nsa & w = (b_{1,1}, \ldots, b_{m,n}, t_{1,1}, \ldots, t_{m,n}, p_1, \ldots, p_m) \in W_{sa} \\cup \{\text{\textbf{start} } i, j \in \{0, 1\} \cup \{p, p_j \in lsa, & i \in Nsa, \} \in Gsa\}$ and all $j \in Gsa$ and all $0 \leq p_r < \text{max}_sa - \text{inc}$ such that either (i) $p_r = 0$ and $t_{i,j} = 0$ or (ii) $p_r = p & t_{i,j} = 0$, for all $i \in Nsa$ or (iii) $p_r = p & t_{i,j} = 1$)
- $U_{sa}$ is defined as: for all $w = (b_{1,1}, \ldots, b_{m,n}, t_{1,1}, \ldots, t_{m,n}, p, p_1, \ldots, p_m) \in W_{sa}$ and all $d \in A_{sa}$
- $\text{if } w \notin T_{sa}, \text{then: } U_{sa}(w, d) = (b'_{1,1}, \ldots, b'_{m,n}, t'_1, \ldots, t'_m, p', p'_1, \ldots, p'_m)$, where for every $i \in Nsa$ and $j \in Gsa$, (i) $b'_{i,j} = 1$ iff $(i, 1, p_{r,1}, \ldots, (1, 1, p_m), 0)$ and $p_r \neq 0$ and $b'_{i,j} = 0$ otherwise; (ii) $t'_{i,j} = 1$ iff $b'_{i,j} = 1$ and for all $r \in Gsa$, $t'_{i,j}$ is 0 otherwise; (iii) $p'_{r} = p + \text{inc}$; (iv) $p'_{r} = p + \text{inc}$ iff $t'_{i,j} = 0$ for all $r \in Gsa$, and $p'_{r} = p_{r}$ otherwise.
- Otherwise, $U_{sa}(w, d) = w$.
- For each $w \in W_{sa}, i \in Nsa$ and $j \in Gsa$, (i) $\pi_{\text{sa}}(w, \text{trade}_{i,j}) = t_{i,j}$; (ii) $\pi_{\text{sa}}(w, \text{price}) = p_j$; (iii) $\pi_{\text{sa}}(w, \text{price}) = p_j$; (iv) $\pi_{\text{sa}}(w, \text{alloc}_{i,j}) = 0$ iff $w \notin T_{sa}$ and $\pi_{\text{sa}}(w, \text{alloc}_{i,j}) = t_{i,j}$ otherwise; (v) $\pi_{\text{sa}}(w, \text{payment}) = \sum_{j \in Gsa} (p_j \pi_{\text{sa}}(w, \text{trade}_{i,j}))$
- For each $w \in W_{sa}, \pi_{\text{sa}}(w) = \{\text{sold} : t_{i,j} = 1 & j \in Gsa & i \in Nsa\}$ and all $d \in A_{sa}$ and $\text{inc} \in lsa + \{0\}$

\textbf{Example 2.} Let $\Sigma_{sa}$, where start = 2, inc = 1 and the agents and good sets are the same from Example 1. Figure 5 illustrates a path in $M_{sa}$. In state $w_0$, agents r1 and r2 bid for good a, but only agent r1 bid for good b. In state $w_1$, since r1 is the only bidder for b, b is sold to her. Agent r1 needs to keep her bid for b and r2 cannot
longer bid for it. In $w_1$, agent r2 increases its bid for good a and agent r1 do not bid for a. In state $w_2$, since r2 is the only bidder for a, she buys the good. Since all the goods were sold, this state is terminal.

Figure 5: A Path in $M_{sa}$, with 2 bidders and 2 goods

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

PROPOSITION 7. $M_{sa}$ is an ST-model and it is a model of $\Sigma_{sa}$.

Proof. (Sketch) It is routine to check that $M_{sa}$ is actually an ST-model. Given a path $s$, any stage $t$ of $M_{sa}$, we need to show that $M_{sa}, \delta, t \models \varphi$, for each $\varphi \in \Sigma_{sa}$. Let us verify Rule 1. Assume $M_{sa}, \delta, t \models \text{initial}$ if $\delta(t) = w_{sa}$. By the definition of $\tilde{w}_{sa}, \tilde{\pi}_{sa}$ and $\pi_{sa}$, we have $\pi_{sa}(\tilde{w}_{sa}, \text{price}) = \text{start}, \pi_{sa}(\tilde{w}_{sa}, \text{price}) = \text{start}, \tilde{\text{bid}}_{i,j} \in \pi_{sa}(\tilde{w}_{sa})$ and $\tilde{\text{trade}}_{i,j} = 0$, for all $i \in N_{sa}$ and $j \in G_{sa}$. Thus, $M_{sa}, \delta, t \models \text{initial}$ if $M_{sa}, \delta, t \models \text{price} = \text{start}$ and $\forall \delta \in G_{sa}(\text{price} = \text{start} \land \forall \in \text{EN}(\neg \tilde{\text{bid}}_{i,j} \land \tilde{\text{trade}}_{i,j} = 0))$.

Now we verify Rule 2. Let $j \in G_{sa}$ be a good type. Assume $M_{sa}, \delta, t \models \tilde{\text{sold}}_{j}$ if $\tilde{\text{sold}}_{j} \in \pi_{sa}(\delta(t))$ and $\pi_{sa}(\delta(t), \text{price}) = \text{start}, \tilde{\text{bid}}_{i,j} \in \pi_{sa}(\tilde{w}_{sa})$ and $\tilde{\text{trade}}_{i,j} = 1$ for some $j \in G_{sa}$. Here, we use Rule 3. Assume $M_{sa}, \delta, t \models \text{terminal}$ if $\delta(t) \neq w_{sa}$ and for all $j \in G_{sa}$, either $M_{sa}, \delta, t \models \text{trade}_{i,j} = 1$ for some $r \in N_{sa}$ or $M_{sa}, \delta, t \models \neg \tilde{\text{bid}}_{i,j}$ for all $i \in N_{sa}$. By Rule 2, $M_{sa}, \delta, t \models \text{terminal}$ if $M_{sa}, \delta, t \models \text{initial} \land \forall \in G_{sa}(\tilde{\text{sold}}_{j} \land \forall \delta \in G_{sa}(\neg \tilde{\text{bid}}_{i,j} \land \tilde{\text{trade}}_{i,j} = 0))$.

Now we verify Rule 10. Let $i \in N_{sa}$ and $j \in G_{sa}$. Assume $M_{sa}, \delta, t \models (\text{does}(OR((1,1,p_1), \ldots, (1,m,p_m)), (\text{price})) = \text{start}. By the update function, for any stage $t$, if $\delta(t) \neq w_{sa}$, then $\pi_{sa}(\delta(t+1), \text{price}) = \text{start}, \pi_{sa}(\delta(t), \text{price}) = \text{start}$. For the sake of contradiction, let us assume $\delta(t)$ is not complete. Let $i \in N_{sa}$ be any agent. By the definition of $\text{legal}_{i}(OR((1,1,p_1), (\text{price})) = \text{start}, \pi_{sa}(\delta(t+1), \text{price}) = \text{start}$.
... (1, m, pn), 0) \in L(\delta[t], i), \text{ for all } 0 \leq p_j < \max_{a} \text{ and } j \in G_{sa}, \text{ such that either (i) } p_j = 0 \& \pi_{Y,sa}(\delta[t], \text{trade}_{e,j}) = 0, \text{ or (ii) } p_j = \text{price} \& \pi_{Y,sa}(\delta[t], \text{trade}_{e,j}) = 0 \text{ for all } r \in N_{sa}, \text{ or (iii) } p_j = \text{price} \& \pi_{Y,sa}(\delta[t], \text{trade}_{e,j}) = 0. \text{ Since } \pi_{Y,sa}(\delta[t+1], \text{price}) > \pi_{Y,sa}(\delta[t], \text{price}), \text{ there will be a stage } e \geq 0 \text{ in } \delta, \text{ where the condition (ii) will not hold true for any } 0 \leq p_j < \max_{a} - \text{inc}. 

Thus, for each good j, it will be the case that if i bids 0 for it or the good was assigned to her (i.e., \pi_{Y,sa}(\delta[e], \text{trade}_{e,j}) = 1). From Rules 3 and 10 in \Sigma_{sa}, it follows that \delta[e+1] \in T_{sa}. \text{ Thus, } \delta \text{ is a complete path, which is a contradiction. □}

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

**Proposition 12.** M_{sa} \not\models BB \text{ and } M_{sa} \models \text{noDeficit}.

**Proof.** (Sketch) Given a path \delta in M_{sa} and a stage t in \delta, let us show a counterexample. Note that each allocation can be either 0 or 1 and the good price is at least 0, i.e., \pi_{Y,sa}(\delta[t], \text{alloc}_{i,j}) \in \{0, 1\} \text{ and } \pi_{Y,sa}(\delta[t], \text{price}) \in L_{sa}. Assume \pi_{Y,sa}(\delta[t], \text{alloc}_{i,j}) = 1 \text{ and } \pi_{Y,sa}(\delta[t], \text{price}) > 0, \text{ for some } j' \in G_{sa} \text{ and some } i' \in N_{sa}. \text{ It follows from Rule 12 that } M_{sa}, \delta, t, \models \text{payment}_{t} > 0 \text{ and } M_{sa}, \delta, t \models \text{add(payment}_{1}, \ldots, \text{payment}_{n}) > 0. \text{ Thus, } M_{sa} \not\models BB \text{ and } M_{sa} \models \text{noDeficit}. □

Finally, the agents can always ensure that their utility will be at least as good in the next state as it was in the current, i.e, IR_{t} is valid in M_{sa}, for each i.

**Theorem 3.** For each i \in N_{sa} and some valuation \delta_{i} over individual trades, M_{sa} \models IR_{i}.

**Proof.** Given a path \delta in M_{sa} and a stage t, let T_{t} = OR(((1, 1, p_{1}), \ldots, (1, m, p_{m})), 0), \text{ where } p_{j} = 0 \text{ if } \pi_{Y,sa}(\delta[t], \text{trade}_{e,j}) = 0; \text{ otherwise } p_{j} = \pi_{Y,sa}(\delta[t], \text{price}), \text{ for each } j \in G_{sa}. \text{ Since } T_{t} \in L_{sa}(\delta[t], i), \text{ we can construct a path } \delta', \text{ such that } \delta'[0, t] = \delta[0, t], \delta'(t, t) = T_{t}, \text{ and } \delta'(t', t) = \delta'(t, t), \text{ for all } r \in N_{sa} \setminus \{t\}. \text{ Thus, } M_{ce}, \delta', t \models \text{utility}_{i}(\text{trade}_{e}, \text{payment}_{i}) = x \rightarrow \text{utility}_{i}(\text{trade}_{e}, \text{payment}_{i}) = x, \text{ for } x \in I_{sa} \text{ and } M_{ce}, \delta, t \models IR_{i}. □

6 DISCUSSION AND CONCLUSION

In this paper, we have presented a unified framework for representing auction protocols. Our work is at the frontier of auction theory and knowledge representation.

**Related work.** Our work is rooted in the key contributions on Combinatorial Auctions [14–16, 22]. All these works adopt a mechanism design perspective: they focus on the properties of a given protocol and bidding language. Our work has a different purpose. The CEDL language includes the bidding part of a protocol, but also the protocol itself. Such a language can be used to automatically derive properties for these protocols. CEDL can also be used as a framework for testing new auction protocols.

To the best of our knowledge, almost all contributions on the computational representation of auction-based markets focus on the implementation of the winner determination problem. For instance, Baral and Uylan (2001) show how a specific auction, namely combinatorial auctions, 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. The closest contributions to ours are the Market Specification Language (MSL) [20] and ADL [15], also based on GDL. They both focus on representing single-side auction through a set of rules and then interpreting an auction-instance with the help of a state-based semantics. MSL is limited to single agent perspective while ADL is not. However, the main limit of both approaches is the absence of a bidding language.

**Going Further.** First direction is Computational complexity. Although the model-checking (MC) problem in ADL is PTIME [13], the winner determination in Combinatorial Auctions (and thus also in Combinatorial Exchange) is known to be NP-complete [18]. We aim to explore how the MC problem in CEDL is affected by these results. Clearly, the fragment of CEDL without formulas referring to the WD mixed-integer program is still PTIME. For instance, the Simultaneous Ascending Auction protocol described in this paper avoids such formulas.

CEDL definitely puts the emphasis on the auctioneer and auction designer. Our second direction is to design a CEDL-based General Auction Player (GAP) that can interpret and reason about the rules of an auction-based market. The key difference, when the players’ perspective is considered, is the epistemic and strategic aspects: players have to reason about other players’ behavior. The epistemic component will allow an agent to bid according to her beliefs about other agents’ private values. Our future GAP should then be based on the epistemic extensions of GDL such as GDL-III [19] and Epistemic GDL [7].

**ACKNOWLEDGMENTS**

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

**REFERENCES**

[1] Chitta Baral and Ceri Uylan. 2001. Declarative Specification and Solution of Combinatorial Auctions Using Logic Programming. In Logic Programming and Non-monotonic Reasoning, Thomas Eiter, Wolfgang Faber, and Mirosław Truszczyński (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg. 168–199.

[2] Craig Boutilier and Holger H. Hoos. 2001. Bidding Languages for Combinatorial Auctions. In Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01), Bernhard Nebel (Ed.). Morgan Kaufmann, Seattle, Washington, USA, 1211–1217.

[3] Ruggiero Cavallo, David C Parkes, Adam I Juda, Adam Kirsch, Alex Kulesza, Sébastien Lahaye, Benjamin Lubin, Loizos Michael, and Jeffery Shneidman. 2005. TBBL: A tree-based bidding language for iterative combinatorial exchanges. In Multidisciplinary Workshop on Advances in Preference Handling, Edinburgh.

[4] Peter Cranston. 2011. Simultaneous Ascending Auctions. American Cancer Society. https://doi.org/10.1002/9780470400531.eorms0783

[5] Michael Gเนerse and Michael Thiel. 2014. General game playing. Morgan & Claypool Publishers.

[6] Andrea Giovannucci, Jesús Cerequides, Ullé Endriss, and Juan A Rodriguez-Aguilar. 2010. A graphical formalism for mixed multi-unit combinatorial auctions. Autonomous Agents and Multi-Agent Systems 20, 3 (2010), 342–368. https://doi.org/10.1007/s10458-009-9085-x

[7] Guifre Jiang, Dongmo Zhang, Laurent Perrussel, and Heng Zhang. 2016. Epistemic GDL: A logic for representing and reasoning about imperfect information games. In IJCAI International Joint Conference on Artificial Intelligence.

[8] Paul Klemperer. 1999. Auction Theory: A Guide to the Literature. Journal of Economic Surveys 13, 3 (1999), 227–286. https://doi.org/10.1111/1467-6419.00083 arXiv:https://arxiv.org/abs/1401.6983

[9] Vijay Krishna. 2009. Auction Theory. Academic Press. https://doi.org/10.1192/ bjp.111.179.1009-a
[10] H. Geun Lee and R. Lee. 1997. A hybrid approach of linear programming and logic modeling for the market core of sealed bid auctions. *Annals of Operations Research* 75 (1997).

[11] Benjamin Lubin, Adam I Juda, Ruggiero Cavallo, Sébastien Lahaie, Jeffrey Shneidman, and David C Parkes. 2008. ICE: An expressive iterative combinatorial exchange. *Journal of Artificial Intelligence Research* 33 (2008), 33–77.

[12] Debasis Mishra and Tridib Sharma. 2018. A simple budget-balanced mechanism. *Social Choice and Welfare* 50, 1 (2018), 147–170.

[13] Munyque Mittelmann and Laurent Perrussel. 2020. Auction Description Language (ADL): a General Framework for Representing Auction-based Markets. In *ECAI 2020*, G. de Giacomo (Ed.). IOS Press, Santiago de Compostela.

[14] Noam Nisan. 2000. Bidding and Allocation in Combinatorial Auctions. In *ACM Conference on Electronic Commerce*. 1–12.

[15] Noam Nisan. 2004. Bidding languages. *Combinatorial Auctions* (2004), 1–19. https://doi.org/10.1109/CCEC.2009.42

[16] David C Parkes, Ruggiero Cavallo, Nick Elprin, Adam Juda, Sébastien Lahaie, Benjamin Lubin, Loizos Michael, Jeffrey Shneidman, and Hassan Sultan. 2005. ICE: An Iterative Combinatorial Exchange. In *Proceedings of the 6th ACM Conference on Electronic Commerce (EC ’05)*. Association for Computing Machinery, New York, NY, USA, 249–258. https://doi.org/10.1145/1064009.1064036

[17] David Christopher Parkes and Lyle H Ungar. 2001. *Iterative combinatorial auctions: Achieving economic and computational efficiency*. University of Pennsylvania, Philadelphia, PA.

[18] Michael H Rothkopf, Aleksandar Pekeč, and Ronald M Harstad. 1998. Computationally manageable combinatorial auctions. *Management science* 44, 8 (1998), 1131–1147.

[19] Michael Thielscher. 2016. GDL-III: A proposal to extend the game description language to general epistemic games. In *Proc. the European Conference on Artificial Intelligence (ECAI)*, Vol. 285. Hague, 1630–1631. https://doi.org/10.3233/978-1-61499-672-9-1630

[20] Michael Thielscher and Dongmo Zhang. 2010. From General Game Descriptions to a Market Specification Language for General Trading Agents. *Springer Berlin Heidelberg*, 259–274.

[21] Mark Voorneveld. 2003. Characterization of Pareto dominance. *Operations Research Letters* 31, 1 (2003), 7 – 11. https://doi.org/10.1016/S0167-6377(02)00189-X

[22] Mu Xia, Jan Stallaert, and Andrew B. Whinston. 2005. Solving the combinatorial double auction problem. *European Journal of Operational Research* 164, 1 (2005), 239–251. https://doi.org/10.1016/j.ejor.2003.11.018

[23] Dongmo Zhang. 2018. *A Logic for Reasoning About Game Descriptions*. In *AI 2018: Advances in Artificial Intelligence*, Tanja Mitrovic, Bing Xue, and Xiaodong Li (Eds.). Springer International Publishing, Cham, 38–50.