Noname manuscript No.
(will be inserted by the editor)

A Graphical Formalism for Mixed Multi-Unit Combinatorial Auctions

Andrea Giovannucci · Jesús Cerquides · Ulle Endriss · Juan A. Rodríguez-Aguilar

Received: date / Accepted: date

Abstract Mixed Multi-Unit Combinatorial Auctions are auctions that allow participants to bid for bundles of goods to buy, for bundles of goods to sell, and for transformations of goods. The intuitive meaning of a bid for a transformation is that the bidder is offering to produce a set of output goods after having received a set of input goods. To solve such an auction the auctioneer has to choose a set of bids to accept and decide on a sequence in which to implement the associated transformations. Mixed auctions can potentially be employed for the automated assembly of supply chains of agents. However, mixed auctions can be effectively applied only if we can also ensure their computational feasibility without jeopardising optimality. To this end, we propose a graphical formalism, based on Petri nets, that facilitates the compact represention of both the search space and the solutions associated with the winner determination problem for mixed auctions. This approach allows us to dramatically reduce the number of decision variables required for solving a broad class of mixed auction winner determination problems. An additional major benefit of our graphical formalism is that

Andrea Giovannucci
SPECS, IUA
Universitat Pompeu Fabra E-mail: agiovannucci@iua.upf.edu

Jesus Cerquides
WAI, Volume Visualization and Artificial Intelligence Research Group
Departament de Matemàtica Aplicada i Anàlisi
Universitat de Barcelona
Dept. de Deporte e Informtica
Universidad Pablo de Olavide
E-mail: cerquide@maia.ub.es

Juan A. Rodríguez-Aguilar
IIIA, Artificial Intelligence Research Institute
CSIC, Spanish National Research Council
E-mail: jar@iiia.csic.es

Ulle Endriss
ILLC, Institute for Logic Language and Computation
University of Amsterdam
E-mail: ulle.endriss@uva.nl

it provides new ways to formally analyse the structural and behavioural properties of mixed auctions.

1 Introduction

A combinatorial auction (CA) is an auction where bidders can buy (or sell) entire bundles of goods in a single transaction [1]. Selling goods in bundles has the great advantage of eliminating the risk for a bidder of not being able to obtain complementary goods at a reasonable price in a follow-up auction. For example, think of a CA for a pair of shoes, as opposed to two consecutive single-item auctions for each of the individual shoes. The study of the mathematical, game-theoretic and algorithmic properties of CAs has recently become a popular research topic in artificial intelligence and multi-agent systems. This is due not only to their relevance to important application areas such as electronic commerce or supply chain management, but also to the range of deep research questions raised by this auction model.

In particular, supply chain formation (SCF) appears as a very promising application area where strong complementarities arise. Indeed, Walsh et al. [2] observe that production technologies often have to deal with strong complementarities: the inputs and outputs of a production process are strongly connected since a producer may risk to produce unsold goods, as well as fail to produce already sold goods when it is unable to obtain the inputs, thus losing credibility on the market. Hence, a supply chain can be regarded as an intricate network of producers (entities transforming input goods into output goods at a certain cost), and consumers interacting in a complex way. Nevertheless, the complementarities arising in SCF are different from the ones we do find in CAs. The complementarities in SCF arise because of the preconditions and post-conditions of production processes: precedences and dependencies along the supply chain must be taken into account. Hence, whilst in CAs the complementarities can be simply represented as relationships among goods, in SCF the complementarities involve not only goods, but also interrelated *transformation* (production) relationships along several levels of the supply chain.

The first attempt to deal with the SCF problem by means of CAs was undertaken by Walsh et al. in [3]. In order to automate SCF, they introduce the notion of *task dependency network* as a way of capturing complementarities among production processes (operations). Although very significant, this work does not allow bidders to express their preferences over bundles of transformations; it does not define a bidding language; and the structure of the supply chain has to fulfil strict criteria (e.g. acyclicity, operations can only produce one output good, etc).

In order to overcome these drawbacks, we introduced in [4] the so-called *mixed multi-unit combinatorial auctions* (henceforth *mixed auction* for short) and discussed the issues of bidding and winner determination. Mixed auctions are a generalisation of the standard model of CAs. Thus, rather than negotiating over goods, in mixed auctions the auctioneer and the bidders can negotiate over *supply chain operations* (henceforth *operations* for short), each one characterised by a set of input goods and a set of output goods. A bidder offering an operation is willing to produce its output goods after having received its input goods along with the payment specified in the bid. While in standard CAs, a solution to the winner determination problem (WDP) is a set of atomic bids to accept that maximises the auctioneer's revenue, in mixed auctions, the *order* in which the auctioneer "uses" the accepted operations matters.

Thus, a *solution* to the WDP is a *sequence of operations*. For instance, if bidder *Joe* offers to make dough if provided with butter and eggs, and bidder *Lou* offers to bake a cake if provided with enough dough, the auctioneer can accept both bids whenever he uses Joe's operation before Lou's to obtain baked cakes from butter and eggs.

Just as the WDP for CAs, the WDP for mixed auctions can also be solved by means of an Integer Linear Program (ILP), as shown in [4]. While this provides a first algorithmic solution to the WDP, in the one proposed in [4] the number of variables grows quadratically with the overall number of operations mentioned in the bids. Hence, such an ILP limits the application of mixed auctions to small and medium scenarios, as empirically shown in [5]. Hence, in order for mixed auctions to be effectively applied to SCF, we must ensure computational feasibility in practice while preserving optimality. In this paper we make headway along this direction by extending our own work in [6].

We provide a graphical formalism that allows to compactly represent the search space of the WDP for mixed auctions, leading to dramatic computational savings. In order to obtain such a formalism we proceed in two steps. Firstly, we define a new type of Petri Nets [7], the so-called *Weighted Transition Petri Nets* (henceforth *weighted nets* for short), to express the notion of transformation (production) cost. Petri nets are a well known graphical tool to analyse discrete dynamical systems. We resort to Petri Nets because: (i) they can naturally help capture the notions of transformation; (ii) they have a well-defined semantics that can naturally accommodate the notion of sequence of transformations; (iii) they have an integrated description of both states and actions to characterise the search space where transformations occur; (iv) they have a large number of formal analysis methods that allow the investigation of their structural and dynamic properties; and (v) they have a graphical representation that is intuitively very appealing to study problems related to the topology of the supply chain. Secondly, we introduce and solve a new type of reachability problem over weighted nets to which we map the mixed auction WDP. Two major benefits, and therefore contributions, stem from this process. First of all, as a main benefit, we do manage to provide a formalism with which mixed auctions, and therefore all auction types subsumed by mixed auctions, in particular CAs for SCF, can be formally analysed. For instance, topological problems of a supply chain can be readily analysed by means of adapting Petri Nets tools. As a second benefit, direct consequence of the provided mapping to weighted nets, we manage to dramatically reduce the number of decision variables in the optimisation problem posed by mixed auctions from quadratic to linear for a wide class of WDPs. Hence, we make headway in the practical application of mixed auctions, and in particular to SCF as intended.

The paper is structured as follows. In Section 2 we provide some background on CAs and discuss some relevant issues in the related literature. Then, in Section 3 we delve into bidding languages and the WDP for mixed auctions. In Section 4 we introduce weighted nets and a new optimisation problem on them: the so-called Constrained Maximum Weighted Occurrence Sequence Problem. Next, we explain how to solve such optimisation problem by means of an ILP. In Section 5 we show how to map a mixed auction WDP to a Constrained Maximum Weighted Occurrence Sequence Problem, thereby managing to significantly reduce the WDP search space. Finally, in Section 6 we draw some conclusions and illustrate some paths to future research.

2 Background and Related work

A *combinatorial auction* (CA) [1] is an auction where bidders can buy (or sell) entire bundles of goods in a single transaction. A CA is *single-unit* when there is a single copy of each item at auction (each item has multiplicity one). We say that a CA is *multi-unit* when there are multiple copies of some item(s). Although CAs are computationally very complex [8], the fact that bidders can express their preferences over bundles of goods may help both bidders and auctioneer to obtain better deals. In fact, buying items in bundles has the great advantage of eliminating the risk for a bidder of not being able to sell complementary items at a reasonable price in a follow-up auction. Indeed, CAs may lead to more efficient allocations whenever complementarities among the goods at auction hold.

CAs can potentially be employed as an allocation mechanism in a wide variety of real-world domains. Thus, they have been proposed to be employed for allocating loads to trucks in the transportation market [9], routes to buses [10], goods/services to buyers/providers in industrial procurement scenarios [11], airport arrival and departure slots [12], radio-frequency spectrum for wireless communications services [13], and supply chain formation [2].

Auction theory studies the formal properties of auctions [14,15]. CAs have recently attracted the attention of economists and game theorists. Associated to auction theory is also the design of auction mechanisms, devoted to study *how* to run an auction in order to guarantee some economic properties such as, for instance, efficiency, incentive compatibility, individual rationality, etc. Regarding mechanism design issues of CAs, the reader can refer to [1].

Bidding is the process of transmitting —truthfully or otherwise— one's valuation function over the sets of goods on auction to the auctioneer. A *bidding language* is a formal language for representing valuations. Ideally, it should allow for a compact representation of the information to be transmitted, and it should allow the auctioneer to reason about this information efficiently. The most widely used bidding language is the so-called OR-language. In this setting, each bidder can submit any number of atomic bids (bundles of goods labelled with a price), and the auctioneer may accept any set of atomic bids, provided the associated bundles do not overlap, and charge each bidder the sum of prices of their accepted bids. In the XOR-language, on the other hand, the auctioneer may accept at most one atomic bid from each bidder. Different langauges can differ in expressive power, compactness of representation, and complexity of the associated reasoning problems. Nisan [16] surveys the formal properties of this family of bidding languages in detail. In [4] we have shown how to adapt these languages to mixed auctions, and in the next section we will recall the definition of the XOR-language in detail.

The winner determination problem (WDP) is the problem of choosing which goods to award to which bidder so as to maximise the auctioneer's revenue. WDP for CAs is a complex computational problem. In fact, one of the fundamental issues limiting the applicability of CAs to real-world scenarios is the computational complexity associated to the WDP. In particular, it has been proved that the WDP for CAs is $\mathcal{NP}$-complete [17]. General ILP solvers [18] and special purpose algorithms (e.g. [19], [20], and [21]) have been employed to solve WDP. For an extended review on WDP and related issues the reader can refer to [22], [23], and [24].

According to Walsh et al. [3], "Supply Chain Formation (SCF) is the process of determining the participants in a supply chain, who will exchange what with whom,

and the terms of the exchanges". CAs are a negotiation mechanism well suited to deal with complementarities among the goods at trade. Since production technologies often have to deal with strong complementarities, the automation of SCF appears as a very promising application area for CAs. However, whilst in CAs the complementarities can be simply represented as relationships among goods, in SCF the complementarities involve not only goods, but also *operations* (production relationships) along several levels of the supply chain. The first attempt to deal with the SCF problem by means of CAs was undertaken by Walsh et al. in [3]. In order to automate SCF, they introduce the notion of task dependency network as a way of capturing complementarities among production processes. Although very significant, this work does not allow bidders to express their preferences over bundles of production processes; it does not define a bidding language; and the structure of the supply chain has to fulfil strict criteria (e.g. acyclicity, processes can only produce one output good, etc). Thus, we observe that further requirements (namely *expressiveness*, *computability*, and *formal analysis*) must be addressed to fully support automated SCF. As to *expressiveness requirements*, we need: to represent complementarities among production processes (e.g. if iron and copper are melt together in the same oven, the transformation can be offered at a lower cost than the service of transforming iron and copper separately); to represent production relationships with multiple output goods (e.g. the quartering of a cow to sell its parts); to offer some bidding language to express combinations of bids; to consider the notion of free disposal (buy goods or transformations that remain unused); to support the specification of the configuration to end up with (a supply chain manager may be interested in finishing the SCF process with a given surplus of goods); to support a wide range of supply chain topologies beyond acyclic nets. As to *computational requirements*, we must ensure computational feasibility of SCF while preserving optimality. Finally, as to *formal requirements*, we advocate for counting on a formalism that supports the formal study of structural and behavioural properties of a supply chain. We argue that our contributions in [4] along with the contributions detailed in this paper allow us to satisfactorily address the requirements above, and therefore address the automation of supply chains of agents.

3 Mixed Multi-unit Combinatorial Auctions

In order to successfully tackle the automation of SCF, we introduced in [4] mixed multi-unit combinatorial auctions (mixed auctions), a generalization of the standard model of CAs. Rather than negotiating over goods, in mixed auctions the auctioneer and the bidders can negotiate over supply chain operations, each one characterized by a set of input goods and a set of output goods.

In this section we review previous work on bidding and winner determination for mixed auctions. In particular, we describe the features of the proposed bidding language, designed to allow bidders to express several types of complex bids; and we summarize the features of a general ILP solver that solves the WDP by finding the optimal sequence of supply chain operations. The material contained in this section is described in [4], and more extensively in [25].

3.1 Bidding Language

Since we deal with supply chain formation, in our case the bidding language must be able to refer to operations across the supply chain. In order to cope with this requirement, we distinguish three kinds of *supply chain operations*:(i) the *supply* of a manufacturing operation; (ii) the *supply* of a bundle of goods; and (iii) the *request* of a bundle of goods. In what follows we use *operation* as an abbreviation of *supply chain operation*.

Definition 1 Let G be the finite set of all the types of goods under consideration. An *operation* is a pair $(\mathcal{I}, \mathcal{O})$ of multisets over G. □

We recall that given a set G, a multiset over G is a function from G to $\mathbb{N}$. It can be seen as a subset of G where the number of appearances of each element is counted. Whenever G is a finite set, finite multisets over G can be identified with non-negative integer vectors of dimension $|G|$. Let $a, b \in G$, by $3'a + 5'b$ we represent a multiset containing three copies of a and five copies of b. If $\mathcal{X} = 6'a + 2'b$, then $\mathcal{X}(a) = 6$ and $\mathcal{X}(b) = 2$ and $\mathcal{X}(g) = 0$ for all g such that $b \neq g \neq a$. In general, given $k \in \mathbb{N} \cup \{0\}$ and $g \in G$, we write the multiset containing k copies of g as $k'g$. Also, given $\mathcal{X}$ and $\mathcal{Y}$ multisets over G, $\mathcal{X} + \mathcal{Y}$ is their union, defined as $(\mathcal{X} + \mathcal{Y})(g) = \mathcal{X}(g) + \mathcal{Y}(g)$ for each good $g \in G$.

An agent offering operation $(\mathcal{I}, \mathcal{O})$ declares that it can deliver $\mathcal{O}$ *after* having received $\mathcal{I}$. For instance, $(\{\,\}, 1'a)$ means that the agent in question is able to deliver a (no input required), and $(1'b, 1'c)$ means that it is able to deliver c if provided with b.

Our goal is to have agents negotiate over bundles of operations. That is, in our setting bidders can offer any number of such operations, including several copies of the same operation. Then, we have to introduce a formalism that allows an agent to express preferences over bundles of operations. That is, agents will be negotiating over *multisets of operations*. Notice that such a formalism will allow the bidder to express:

- *offers for bundles of goods*, expressed as operations with no inputs. That means that nothing is taken as input ($\mathcal{I} = \{\,\}$), and $\mathcal{O}$ is provided as output.
- *requests of bundles of goods*, expressed as operations with no output. That means that $\mathcal{I}$ is taken as input, and nothing ($\mathcal{O} = \{\,\}$) is provided as output.
- *offers for bundles of operations*, expressed as:

$$\{\alpha'_1(\mathcal{I}_1, \mathcal{O}_1) + \alpha'_2(\mathcal{I}_2, \mathcal{O}_2) + \ldots + \alpha'_m(\mathcal{I}_m, \mathcal{O}_m)\}$$

  where $\alpha_i \in \mathbb{N}$ is the multiplicity of operation $(\mathcal{I}_i, \mathcal{O}_i)$.

A bidder can express such preferences by a valuation function. In what follows we provide a definition of valuation.

Definition 2 A *valuation* v is a (typically partial) function from multisets of operations to the real numbers. □

Intuitively, given a multiset of operations $\mathcal{D}$, $v(\mathcal{D}) = p$ means that the agent equipped with valuation v is willing to make a payment of p in return for being allocated all the operations in $\mathcal{D}$ (in case p is a negative number, this means that the agent will accept the deal if it *receives* an amount of $|p|$). For instance, $v(1'(1'oven + 1'dough, 1'oven + 1'cake)) = -20$ means that a bidder can produce a cake for 20 if given an oven and some dough, and that she will return the oven again afterwards.

In order to run the auction, the bidders need to communicate their valuation functions to the auctioneer and to do that they need to use a bidding language. A suitable *bidding language* should allow a bidder to encode choices between alternative bids and the like [16]. Informally, an OR-combination of several bids means that the bidder would be happy to accept any number of the sub-bids specified, if paid the sum of the associated prices. An XOR-combination of bids expresses that the bidder is prepared to accept at most one of them. For the formal definition of the WDP below, we restrict ourselves to bids in the XOR-language, which is known to befully expressive for mixed auctions [4]. In the XOR-language each bidder submits a single bid that is an XOR combination of a set of atomic bids. In order to properly define the XOR-language we need to introduce the concept of atomic bid and what it means to XOR-combine a set of atomic bids.

An *atomic bid* is the smallest piece of information a bidder can submit. It is composed of a finite multiset of operations and a price. An atomic bid $Bid = \text{BID}(\mathcal{D}, p)$ communicates to the auctioneer that the bidder is willing to pay p for being allocated all the operations in $\mathcal{D}$ (or some other operation requiring him to produce at most the same output goods from at least the same input). Hence, it defines the following valuation:

$$v(\mathcal{D}') = \begin{cases} p & \text{if } \mathcal{D} \text{ provides equal or better operations than } \mathcal{D}' \\ -\infty & \text{otherwise} \end{cases}$$

where we say that $\mathcal{D}$ provides equal or better operations than $\mathcal{D}'$ if we can establish a one to one mapping f from $\mathcal{D}$ to $\mathcal{D}'$ and for every operation $op \in \mathcal{D}$, we have that $f(op) \in \mathcal{D}'$ takes at least the same inputs and provides at most the same outputs[1].

Let $Bid = Bid_1 \text{ XOR } \cdots \text{ XOR } Bid_n$ be an XOR-combination of n atomic bids Bid_i, with $i \in \{1, \ldots, n\}$. When a bidder submits it, it is communicating to the auctioneer the following valuation:

$$v(\mathcal{D}) = \max\{v_i(\mathcal{D}) \mid i \in \{1, \ldots, n\}\}$$

That is, the XOR-combination offers the auctioneer the possibility to select the atomic bid maximizing its revenue.

3.2 Winner Determination Problem. Informal Definition

The *input* to the WDP consists of a complex bid expression for each bidder, a multiset $\mathcal{U}_{in}$ of goods the auctioneer holds to begin with, and a multiset $\mathcal{U}_{out}$ of goods the auctioneer expects to end up with.

In standard CAs, a solution to the WDP is a set of atomic bids to accept. In our setting, however, the *order* in which the auctioneer "uses" the accepted operations matters. For instance, if the auctioneer holds a to begin with, then checking whether accepting the two bids $Bid_1 = \text{BID}(1'(1'a, 1'b), 10)$ and $Bid_2 = \text{BID}(1'(1'b, 1'c), 20)$ is feasible involves realising that we have to use Bid_1 before Bid_2. Thus, a *solution* for WDP will be a *sequence of operations*. A *valid* solution has to meet two conditions:

1. *Bidder constraints.* The multiset of operations in the sequence has to *respect the bids* submitted by the bidders, namely:

[1] This "free disposal" condition is treated more formally in [4]. For our main purposes in this paper, the improvement of winner determination algorithms, this is not a relevant detail.

(a) If a bidder submits an offer over a bundle of *operations*, all of them must be employed in the operation sequence.
(b) If a bidder submits an XOR-combination of atomic bids, at most one of them may be accepted.

2. *Auctioneer constraints.* The sequence of operations has to be *implementable*, namely:
(a) the set of goods held by the auctioneer prior each operation must be a superset of the inputs of the operation; and
(b) the set of goods held by the auctioneer at the end of the sequence must be a superset of $\mathcal{U}_{out}$.

An *optimal* solution is a valid solution that maximises the sum of prices associated with the atomic bids selected.

3.3 Solving the WDP by Integer Linear Programming

We now show how to map the WDP defined in Section 3.2 into a more formal definition, by encoding it in ILP. Here and in what follows:

- let n be the total number of bidders;
- i acts as a bidder index and when quantified it ranges over all bidders;
- for each bidder i, we use j as an atomic bid index ranging from 1 to m_i, the number of atomic bids submitted by bidder i;
- $Bid_{ij} = \text{BID}(\mathcal{D}_{ij}, p_{ij})$ is the j-th atomic bid within the XOR-bid submitted by bidder i, $\mathcal{D}_{ij}$ being a multiset of operations and p_{ij} the price of the bid;
- $B = \{Bid_{ij} : 1 \leq i \leq n, 1 \leq j \leq m_i\}$ is the set that contains all the atomic bids;
- for each atomic bid j of bidder i, k acts as an operation index and when quantified ranges from 1 to the number of operations in that atomic bid;
- t_{ijk} is the k-th operation appearing in the j-th atomic bid of bidder i;
- $\mathcal{I}_{ijk}$ and $\mathcal{O}_{ijk}$ are respectively the input and output multisets of operation t_{ijk};
- $\mathcal{D}_{ij}(t_{ijk})$ is the multiplicity of t_{ijk} in $\mathcal{D}_{ij}$;
- $\mathcal{D} = \biguplus_{ij} \mathcal{D}_{ij}$ stands for the multiset of the overall operations received with their multiplicity;[2]
- $T = \{t_{ijk} : \forall ijk\}$ is the set of the overall operations in the bids disregarding their multiplicity;
- δ is the overall number of operations mentioned anywhere in the bids (taking account of their multiplicity); i.e. $\delta = \sum_{ij} |\mathcal{D}_{ij}| = \sum_{ijk} \mathcal{D}_{ij}(t_{ijk})$, and therefore it also stands for the maximum length of the solution sequence (provided all operations are used);
- G is the set types of negotiated goods;
- g ranges over all goods; and
- m always ranges from 1 to δ.

First, as in standard combinatorial auctions, we need to encode which bids are selected. Thus, we define a set of binary decision variables $x_{ij} \in \{0, 1\}$, where x_{ij} takes on value 1 iff the j-th atomic bid of bidder i is selected. Furthermore, for each operation in a selected bid we need to choose its position in the solution sequence. Thus, we define a set of binary decision variables $x^m_{ijk} \in \{0, 1\}$, where x^m_{ijk} takes on value 1 if the operation t_{ijk} is selected at the m-th position of the solution sequence, and 0 otherwise.

[2] The operator $\biguplus$ performs the sum (as described after Definition 1) of several multisets.

In what follows, we define the set of constraints that the solution sequence must fulfil:

1. For simplicity, we impose that at most one operation is selected at each position of the sequence:
$$\sum_{ijk} x_{ijk}^{m} \leq 1 \qquad (\forall m) \tag{1}$$
2. We enforce the constraints expressed by condition (1.a) in Section 3.2. Thus, if bid Bid_{ij} is selected, all the operations t_{ijk} in that bid must be selected exactly $\mathcal{D}_{ij}(t_{ijk})$ times. In other words, if bid Bid_{ij} is selected, all the operations in it must be selected with the required multiplicity. Formally,
$$x_{ij} \cdot \mathcal{D}_{ij}(t_{ijk}) = \sum_{m} x_{ijk}^{m} \qquad (\forall ijk) \tag{2}$$
3. We enforce that the atomic bids submitted by each bidder are exclusive (XOR). This amounts to satisfying the following constraints (cf. condition (1.b) in Section 3.2):
$$\sum_{j} x_{ij} \leq 1 \quad (\forall i) \tag{3}$$
Observe that in the case of the *OR* bidding language we simply have to remove this constraint.
4. Next, we capture condition (2.a) in Section 3.2 requiring that enough goods are available at step m to perform the next operation, namely:
$$\mathcal{U}_{in}(g) + \sum_{l=0}^{m-1} \sum_{ijk} x_{ijk}^{l} \cdot [\mathcal{O}_{ijk}(g) - \mathcal{I}_{ijk}(g)] \geq \sum_{ijk} x_{ijk}^{m} \cdot \mathcal{I}_{ijk}(g) \qquad (\forall g, \forall m) \tag{4}$$
5. And finally, after having performed all the selected operations, the set of goods held by the auctioneer must be a superset of the final goods $\mathcal{U}_{out}$ (cf. condition (2.b) in Section 3.2):
$$\mathcal{U}_{in}(g) + \sum_{m=0}^{\delta} \sum_{ijk} x_{ijk}^{m} \cdot [\mathcal{O}_{ijk}(g) - \mathcal{I}_{ijk}(g)] \geq \mathcal{U}_{out}(g) \qquad (\forall g) \tag{5}$$

Now solving the WDP for MMUCAs with XOR-bids amounts to solving the following binary linear program, which we name Direct Integer Program (DIP):

$$\max \sum_{ij} x_{ij} \cdot p_{ij} \quad \text{subject to constraints (2)– (5)} \tag{6}$$

Now it is time to assess the number of decision variables used by DIP.

Proposition 1 *The number of decision variables used by DIP is quadratic in the number of operations involved in the auction.*

Proof DIP has $\delta \cdot |T| + |B|$ binary decision variables:

- $\delta \cdot |T|$ variables (x_{ijk}^{m}), where δ is the maximum length of the solution sequence and $|T|$ is the size of the set of different operations.
- $|B|$ variables (x_{ij}), where $|B|$ is the number of atomic bids submitted by the bidders.

Hence, the number of decision variables is $\mathcal{O}(\delta \cdot |T|)$. □

Observe that our proposed implementation can easily be amended so as to directly encode the constraints imposed by language constructs other than the XOR-operator.

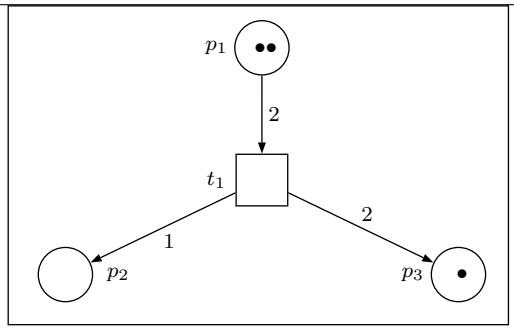

Fig. 1 Example of a Place Transition Net.

4 Extending Place Transition Nets

Our endeavour hereafter will focus on reducing the number of decision variables required by an ILP to solve a mixed auction WDP. Along this path we start in this section by introducing a new optimisation problem on an extension of Place Transition Nets. In Section 5 we show how instances of the WDP for mixed auctions can be transformed into instances of the optimisation problem described in this section. As a result of such transformation we manage to reduce the number of decision variables from quadratic to linear in the number of operations.

In this section, we first introduce Place Transition Nets to subsequently extend them to incorporate a value function. We call the resulting model *Weighted Place Transition Nets*. Afterwards, we define a new optimization problem on Weighted Place Transition Nets, the Constrained Maximum Weighted Occurrence Sequence Problem. Finally, we explain how to solve such an optimization problem (in some special cases) by means of Integer Linear Programming.

4.1 Petri Nets and Place Transition Nets

Petri Nets are a powerful mathematical and graphical tool for the description of discrete distributed systems. They were first introduced in 1962 by Karl Adam Petri in his seminal dissertation [26]. In particular they are suitable for describing systems in which parallelism, concurrency, and synchronization play an important role. We refer the reader to [7] for a very good review.

We will focus on a particular type of Petri net called Place Transition Net. In the following we refer to Place Transition Net as *net* for short. Formally, following [7]:

Definition 3 (Place Transition Net Structure) A *Place Transition Net Structure* is a tuple $N = (P, T, A, E)$ where P is a set of *places*, T is a finite set of *transitions* such that $P \cap T = \emptyset$, $A \subseteq (P \times T) \cup (T \times P)$ is a set of *arcs*, and $E : A \to \mathbb{N}^+$ is an *arc expression* function ($E(t, p)$ stands for the number of tokens introduced by transition t into place p and $E(p, t)$ stands for the number of tokens substracted from place p by transition t). □

In the following, we use the term *structure* to refer to a *Place Transition Net Structure* for short.

A distribution of tokens over the set of places is called a *marking*, and it stands for the state of the net.

Definition 4 (Marking) A *marking* $\mathcal{M} : P \to \mathbb{N}$ of a structure is a multiset over P. $\mathcal{M}(p) = k$ means that place $p \in P$ contains k tokens for marking $\mathcal{M}$. □

A net is a structure S together with a given initial marking $\mathcal{M}_0$. An example of net is shown in Figure 1. The graphical representation of a net is composed of the following graphical elements: places (drawn as circles), transitions (drawn as rectangles), arcs (connecting places to transitions or transitions to places) labelled with values of an arc expression function E. Tokens are drawn as bullet points inside the circles representing places.

Given a marking $\mathcal{M}$, we say that a transition is *enabled* if all its input places contain at least as many tokens as required by the the transition's input arcs. Intuitively, a transition is enabled if enough tokens are present in its input places. In other words, a transition $t \in T$ is said to be *enabled* if each input place p of t is marked with at least $E(p,t)$ tokens, where $E(p,t)$ represents the weight of the arc connecting p to t. More formally,

Definition 5 (Enabled Transition) Given a marking $\mathcal{M}$, a transition $t \in T$ is enabled iff $\mathcal{M}(p) \geq E(p,t) \quad \forall (p,t) \in A$. □

If a transition is enabled it can *fire* consuming the tokens specified by E of the input places and producing the tokens specified by E in the output places. An enabled transition may or may not fire. If it fires, it changes the current marking to a new marking by removing tokens from the input places and putting tokens into the output places. More formally:

Definition 6 (Firing of an enabled transition) The *firing* of an enabled transition t removes $E(p_i,t)$ tokens from each input place p_i and adds $E(t,p_o)$ tokens to each output place p_o. The firing of a transition t changes marking $\mathcal{M}_{k-1}$ to a marking $\mathcal{M}_k$. The new marking can be computed employing the following equation:

$$\mathcal{M}_k(p) = \mathcal{M}_{k-1}(p) + E(t,p) - E(p,t) \quad \forall p \in P \tag{7}$$

Note that in this equation and henceforth, for simplicity, we assume that $E(p,t) = 0$ if $(p,t) \notin A$ and $E(t,p) = 0$ if $(t,p) \notin A$. □

4.1.1 Reachability

Recall that any WDP for a mixed auction defines an initial multiset of goods $\mathcal{U}_{in}$ and a final multiset of goods $\mathcal{U}_{out}$. Hence, we can think of the problem as reaching $\mathcal{U}_{out}$ starting from $\mathcal{U}_{in}$. There is a similar problem in nets: the problem of *reachability*. It studies whether we can reach a particular state of a net departing from a given initial state only by using a sequence of net transitions.

Definition 7 (Firing Sequence) Given a structure S and a marking $\mathcal{M}_0$, a *firing sequence* is a sequence of transitions $\langle t_1,\ t_2, \ldots, t_d \rangle$ from S such that t_1 is enabled in $\mathcal{M}_0$ and for all $i \in [2,d]$, t_i is enabled after firing t_{i-1}. □

By the repeated application of equation (7) a firing sequence $J = \langle t_1,\ t_2, \ldots, t_d \rangle$ brings marking $\mathcal{M}_0$ to $\mathcal{M}_d$:

$$\mathcal{M}_d(p) = \mathcal{M}_0(p) + \sum_{t \in J} [E(t,p) - E(p,t)] \qquad \forall p \in P \tag{8}$$

Definition 8 (Reachability) A marking $\mathcal{M}_d$ is *reachable* from a marking $\mathcal{M}_0$ in a structure S if there exists a firing sequence that transforms $\mathcal{M}_0$ into $\mathcal{M}_d$. □

All the markings reachable from $\mathcal{M}_0$ in a structure S are written $R(S, \mathcal{M}_0)$, and are called the *reachable set* of a net.

4.1.2 Reachability and the state equation

In the general case Lipton [27] proved that the reachability problem is EXPSPACE-hard. And yet, it is reasonable to wonder whether there are special classes of nets on which reachability is computationally simpler. In this section we show that indeed computation is simpler for a particular topology of nets by relying on results proved in [7].

Given a net with r transitions and n places, its *incidence matrix* $A = [a_{ij}]$ is an $r \times n$ matrix of integers. Each entry is given by $a_{ij} = E(t_i, p_j) - E(p_j, t_i)$. Hence, a_{ij} is the difference in the number of tokens in place j when transition i fires once.

We can represent a marking $\mathcal{M}_k$ as an $n \times 1$ column vector M_k such that the j-th entry of M_k represents the number of tokens present in place p_j. We can identify each transition t_i with a *firing vector* u^{t_i}, an $r \times 1$ column vector having a 1 at position i and zeros elsewhere. We can now express equation (7) in matrix form as:

$$M_k = M_{k-1} + \mathbf{A}^T u^t \tag{9}$$

where $\mathbf{A}^T$ stands for the transpose of incidence matrix A.

Given a firing sequence $J = \langle t_1, \ldots, t_d \rangle$, its *firing count vector* K_J is an $r \times 1$ column vector containing at the i-th position the number of times that the i-th transition in T is repeated in sequence J. Formally, $K_J = \sum_{k=1}^{d} u^{t_k}$.

Say that $\mathcal{M}_d$ is reachable from $\mathcal{M}_0$, then there exists a firing sequence $J = \langle t_1, t_2, ..., t_d \rangle$ bringing from marking $\mathcal{M}_0$ to $\mathcal{M}_d$. Therefore, a *necessary condition on reachability* can be expressed in terms of a matrix equation:

Theorem 1 (Murata,1989) *If $\mathcal{M}_d$ is reachable from $\mathcal{M}_0$, then the following equation has a non-negative integer solution* $\mathbf{x}$*:*

$$M_d = M_0 + \mathbf{A}^T \mathbf{x} \tag{10}$$

The proof consists in showing that $\mathbf{x} = K_J$ is a solution. Notice that the i-th entry of vector $\mathbf{x}$ encodes the number of times a transition t_i must be fired to transform $\mathcal{M}_0$ into $\mathcal{M}_d$.

Equation (10) is called the *state equation*, since it describes the states that a net would reach if the transitions encoded in $\mathbf{x}$ were fired. However, notice that the condition is not sufficient because the existence of a solution does not always imply that $\mathcal{M}_d$ is reachable from $\mathcal{M}_0$. However, we can show that sometimes all the states reachable by a net are described by the state equation. In particular, this happens when the net is acyclic.

Definition 9 (Acyclicity) A *directed cycle* in a structure (P, T, A, E) is a sequence of places and transitions $\langle p_1, t_1, p_2, t_2, \ldots, t_{n-1}, p_n \rangle$ such that $p_1 = p_n$ and for all $i \in [1, n]$ we have that $(p_i, t_i) \in A$ and $(t_i, p_{i+1}) \in A$.
A structure is said to be *acyclic* if it does not contain any directed cycle. □

In [7], it is shown that in an *acyclic* net, the condition expressed by Theorem 1 is not only necessary, but also sufficient to guarantee reachability.

Theorem 2 (Murata,1989) *In an acyclic structure* (P, T, A, E)*,* $\mathcal{M}_d$ *is reachable from* $\mathcal{M}_0$ *iff the following equation has a non-negative integer solution in* $\mathbf{x}$*:*

$$\mathcal{M}_d = \mathcal{M}_0 + \mathbf{A}^T\mathbf{x} \tag{11}$$

That is, for any acyclic net, if there exists a solution to equation (11), a firing sequence reaching $\mathcal{M}_d$ from $\mathcal{M}_0$ is guaranteed to exist, and $\mathbf{x}$ represents its firing count vector.

Moreover, Murata further extends the class of nets for which the condition is still sufficient. These particular nets (trap-circuit and syphon-circuit nets) have special topologies with particular types of circuits. For such nets, the state equation represents all the reachable states if the initial marking $\mathcal{M}_0$ satisfies some constraints. Further efforts have been made for extending the validity of the state equation to more classes of nets [28].

4.2 Weighted Place Transition Nets

There is a feature of some discrete systems (in particular the ones we consider) that, to the best of our knowledge, has never been considered so far in the Petri net literature, and that we deem as fundamental. A change in the state of a system may have an associated value. For instance, in our case, an operation has an associated value every time it is performed. Thus, in order to model operations, we need to extend nets to incorporate the notion of *transition value*. Such extension will allow us not only to represent the fact that a value is associated with each transition firing, but also to easily compute the value of a *firing sequence*.

We extend the notion of net (see Section 4.1) by attaching a *value* to each transition. This leads us to the definition of *Weighted Place Transition Net Structure* and *Weighted Place Transition Net* (henceforth referre to as *weighted structure* and *weighted net* respectively for short).

Definition 10 (Weighted Structure) A weighted structure is a a tuple (P, T, A, E, V) where:

- (P, T, A, E) is a structure.
- $V : T \to \mathbb{R}$ is a function that assigns a value to each transition.

□

We define a weighted net by associating to a weighted structure an initial marking $\mathcal{M}_0$.

The initial marking in a net represents the initial state of the system, the very same semantics is inherited by weighted nets.

Weighted structures and weighted nets preserve all the properties of structures and nets respectively, but allow the quantitative representation of the value of a transition. Therefore, we can naturally apply to them all the concepts employed for nets. Those include the concepts of enabling of a transition, firing of a transition, marking, firing sequence, and so on.

A weighted net processes a firing sequence (Section 4.1.1)as follows: firstly, the weighted net evolves through a sequence of markings (states); and secondly, a value is assessed for the firing sequence according to the following definition:

Definition 11 (Value of a firing sequence) The value V of a firing sequence $J = \langle t_1, t_2, ..., t_d \rangle$ is the sum of the values of each transition contained in the sequence:

$$V(J) = \sum_{i=1}^{d} V(t_i) \tag{12}$$

If a transition fires more than once, say k times, then its value will be added k times. □

4.2.1 Constrained Maximum Weight Occurrence Sequence Problem

Since each transition has a value, it is reasonable to wonder about the optimal firing sequence leading from an initial marking to some final marking. Moreover, and more generally, we may require to compute the optimal firing sequence leading from an initial marking $\mathcal{M}_0$ to a final marking $\mathcal{M}_d$ that fulfils some constraints. For instance, a given problem may require that each place in the final marking contains at least one token ($\mathcal{M}_d(p) > 1, \forall p \in P$). In order to model this more general optimisation problem, we define the *Constrained Maximum Weight Occurrence Sequence Problem* (MAXSEQ).

Definition 12 (MAXSEQ) Given

- a weighted net $N = (P, T, A, E, \mathcal{M}_0, V)$,
- a set of inequality constraints over a subset ($P^{\geq}$) of the places of a marking $\mathcal{M}_d$, expressed as:
$$\forall p \in P^{\geq} \qquad \mathcal{M}_d(p) \geq g_p, \tag{13}$$
- and a set of equality constraints over a subset ($P^{=}$) of the places of a marking $\mathcal{M}_d$, expressed as:
$$\forall p \in P^{=} \qquad \mathcal{M}_d(p) = h_p \tag{14}$$

find a firing sequence $J_{opt} = \langle t_1, t_2, ..., t_d \rangle$ that maximizes the sequence value V among the ones that bring from some initial marking $\mathcal{M}_0$ to a final marking $\mathcal{M}_d$ that fulfills the constraints in equations (13) and (14). □

4.2.2 Reducing MAXSEQ to ILP

An ILP takes the form: maximize $f(\mathbf{x})$ subject to $g(\mathbf{x}) \geq \mathbf{g}$ and $h(\mathbf{x}) = \mathbf{h}$ where f,g and h are linear functions, $\mathbf{g}$ and $\mathbf{h}$ are real vectors and $\mathbf{x}$ is a vector of integer variables. Our aim now is to encode MAXSEQ as an ILP. Since the structure is very similar, we only need to identify what ILP variables we will use and then show that the function to maximize and the constraints in MAXSEQ are a linear function of these variables.

As state space we will use a vector $\mathbf{x}$ of r integer variables, r being the number of transitions in the network. Each variable represents the number of times a transition has fired. In Section 4.1.2, we showed that under some hypothesis on a net, it is possible to express its overall reachability set by means of an equation, the state equation. Concretely, Theorem 2 establishes that given an $\mathcal{M}_0$ and a acyclic net with incidence matrix $\mathbf{A}$, the set $\{\mathcal{M}_0 + \mathbf{A}^T\mathbf{x} \geq \mathbf{0} \mid \mathbf{x}$ *is a vector of non-negative integers*$\}$ contains exactly the markings reachable from $\mathcal{M}_0$. This result allows us to represent the MAXSEQ search space by a means of the vector of integers $\mathbf{x}$. Furthermore, this

result also shows that the final marking $\mathcal{M}_d$ in term of which the constraints are expressed, is a linear function of $\mathbf{x}$ (since $\mathcal{M}_d = \mathcal{M}_0 + \mathbf{A}^T\mathbf{x}$). This result can be extended to weighted nets without changes and is the basis for our ILP program.

Notice also that the function to maximize, (that is, the value of each firing sequence) does not depend on the order of the sequence (see equation (12)). Furthermore, since a value is associated to each transition in the net, the value depends linearly on the number of times each transition fires. We define a vector $\mathbf{v} \in \mathbb{R}^{|T|}$ whose j-th position represents the value associated with transition t_j ($v[j] = V(t_j)$). Hence, the value associated with the firing sequence represented by $\mathbf{x}$, is:

$$V(\mathbf{x}) = \mathbf{v}^T\mathbf{x} \tag{15}$$

Now that we have shown the connection between MAXSEQ and ILP, we can formally state it.

Theorem 3 *Given a MAXSEQ instance formed by a weighted net $(P, T, A, E, \mathcal{M}_0, V)$ with incidence matrix $\mathbf{A}$ and the constraints appearing in equations* (13) *and* (14)*:*

If the state equation describes all the reachable states of the weighted net, then all the non-negative integer solutions $\mathbf{x}$ of the following ILP:

$$\begin{aligned}
&\textit{max} && \mathbf{v}^T\mathbf{x} && (16)\\
&\textit{subject to} && \forall p \in P^{\geq} \quad \mathcal{M}_d(p) \geq g_p && (17)\\
& && \forall p \in P^{=} \quad \mathcal{M}_d(p) = h_p && (18)\\
&\textit{where} && \mathcal{M}_d = \mathcal{M}_0 + \mathbf{A}^T\mathbf{x} && (19)
\end{aligned}$$

represent the firing count vectors of all the optimal solutions to the MAXSEQ instance.

Proof Notice that equation (19) computes the final marking as a linear function of $\mathbf{x}$. Since the state equation describes all the reachable states, $\mathcal{M}_d$ will range over every possible final marking. Then equations (17) and (18) directly translate the constraints in equations (13) and (14) in MAXSEQ. Finally, equation (16) computes the value of the firing sequence as given by (15). As a result, a solution $\mathbf{x}^*$ to the ILP defined by equations (16)-(19) optimizes the sum of the values associated with fired transitions while ensuring that the final marking is reachable and fulfils the constraints defined by the MAXSEQ instance. □

Note that the number of integer variables required to solve MAXSEQ via ILP is exactly $|T|$, that is, the number of transitions of the net.

According to the results stated in Theorem 2, it is possible to express the reachability set with the state equation when the net is acyclic. Then, we apply this result to MAXSEQ via the following corollary:

Corollary 1 *Provided that a net is acyclic, every MAXSEQ defined on it can be reduced to ILP.*

Proof Since the net is acyclic, in virtue of Theorem 2, all the reachable states $\mathcal{M}$ are the non-negative integer solutions of equation (11). Given a MAXSEQ instance over that net, by Theorem 3 the firing count vectors of all the solutions are represented by the solutions to the ILP in equations (16)-(19).

Hence, every MAXSEQ instance over the net can be solved in two steps. First, we determine the optimal firing count vector $\mathbf{x}^{opt}$ by solving the ILP in equations (16)-(19). Then, since the net is acyclic, we can establish a partial order among transitions so that $t_1 \prec t_2$ iff t_2 uses as input some output of t_1. We can construct an occurrence sequence J_{opt} by ordering the transitions in $\mathbf{x}^{opt}$ non-decreasingly according to our partial ordering. In that way, every step in J_{opt} is guaranteed to be enabled and consequently J_{opt} is a solution to the MAXSEQ instance. □

5 Mapping mixed auctions to weighted nets

In this section we demonstrate that an instance of the mixed auction WDP can be transformed into an instance of the MAXSEQ problem introduced in Section 4.2.1. Notice that this mapping allows us to benefit from analysis methods to study behavioral properties of Petri nets. Hence, we exploit such analysis methods to provide an ILP formulation for some classes of weighted nets, and therefore some types of supply chain networks.

5.1 Intuitions behind the mapping

The idea behind the mapping is that an atomic operation can be regarded as a transition in a weighted net. Consider the following offer, expressed by a bidder in the bidding language introduced in Section 3.1:

$$Bid_1 = \text{BID}(1'(2'H_2O, 1'O_2 + 2'H_2), -8) \tag{20}$$

This represents an offer to perform an hydrolysis process: 2 moles of water are transformed into 1 mole of oxygen and two moles of hydrogen at a price of €8. Now consider the transition depicted in Figure 2, and say that each place represents a good. Let the place labelled with H_2O be water, H_2 be hydrogen, and O_2 be oxygen. The transition in Figure 2 perfectly captures the semantics of a supply chain operation: the input places of the transitions are the input goods of the operation, its output places are the output goods of the operation, and the transition value is the value associated with the operation. Analogously, an operation offering goods can be represented as a transition with only output places, whereas an operation requesting goods as a transition with only input places.

Example 1 Say that the following bids are submitted to a mixed auction:

$$bid_1 = \text{BID}(1'(\{\}, 2'H_2O), -10) \tag{21}$$

$$bid_2 = \text{BID}(1'(\{\}, 2'H_2O), -14) \tag{22}$$

$$bid_3 = \text{BID}(1'(2'H_2O, 1'O_2 + 2'H_2), -8) \tag{23}$$

$$bid_4 = \text{BID}(1'(2'H_2 + 1'O_2, \{\}), 23) \tag{24}$$

$$bid_5 = \text{BID}(1'(2'H_2 + 1'O_2, \{\}), 25) \tag{25}$$

We can represent them graphically by the weighted net in Figure 3. □

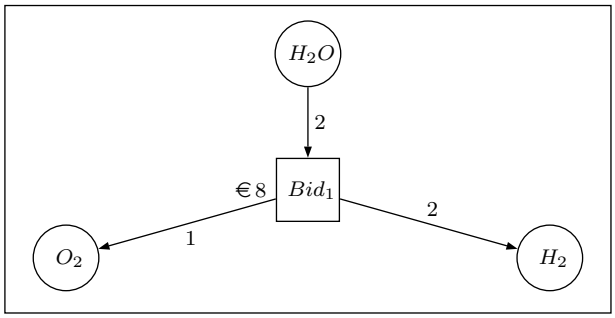

Fig. 2 Example of an operation represented as a transition in a weighted net.

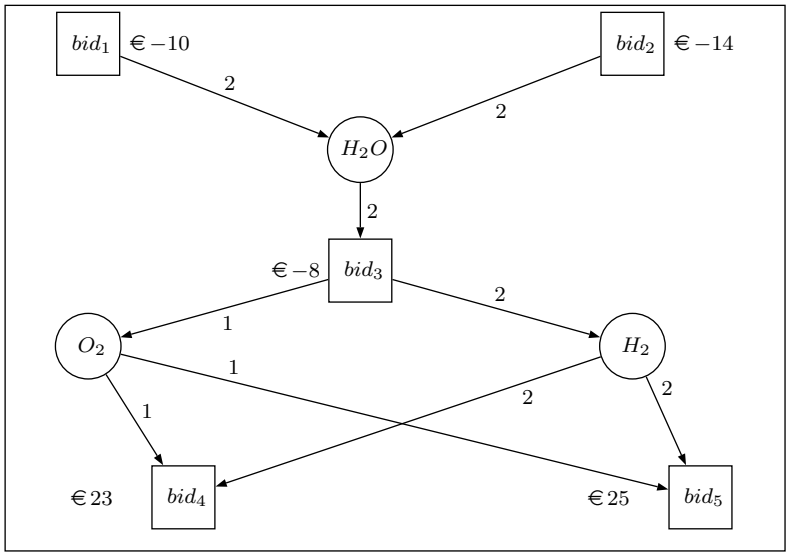

Fig. 3 Example of bids in a mixed auction represented as a weighted net.

Finding the revenue-maximizing solution in example 1 is straightforward. Firstly, buy two moles of water from bid_1, then process the water through the operation in bid_3, and then sell the products of the reaction to bid_5. The total revenue of the supply chain is $25 - 8 - 10 =$€7. Notice that this is also the solution to the MAXSEQ problem[3] defined on the weighted net in Figure 3 with an empty initial marking and with a destination marking $\mathcal{M}_d$ satisfying the following constraints:

$$\mathcal{M}_d(H_2O) \geq 0 \tag{26}$$
$$\mathcal{M}_d(O_2) \geq 0 \tag{27}$$
$$\mathcal{M}_d(H_2) \geq 0 \tag{28}$$

Given the example above, we argue that if we construct a MAXSEQ instance by:

1. building a weighted net joining all the atomic operations received within bids;

[3] So far under the hypothesis that transitions can fire at most once. We will solve the issue of limiting the number of times each transition can fire further on.

2. setting the initial marking to the goods initially available to the auctioneer ($\mathcal{U}_{in}$); and
3. setting some constraints on the final marking ($\mathcal{U}_{out}$),

then the solution to the MAXSEQ instance corresponds to the solution to the mixed auction WDP.

Nonetheless, we must take some more details into account. Firstly, in the previous example, given the weighted net representation, each operation can be used an arbitrary number of times. Instead, the semantics of the bidding language imposes that operations must be used a limited number of times. Secondly, we must provide bidders with the capability of encoding on the weighted net both offers or requests over bundles of operations. Moreover, they also require the capability of encoding on the weighted net sets of mutually exclusive (XOR) atomic bids. Addressing these three issues is the purpose of the following section.

5.2 Representing Bids

In Example 1, we restricted ourselves to the case in which agents can only submit one atomic bid. Moreover, we only consider bidding over a single atomic operation, i.e. $|\mathcal{D}_{ij}| = 1$. Next, we progressively relax all these constraints. First of all, we explain how to represent a weighted net a bid on a bundle (multiset) of operations.

5.2.1 Expressing bids on bundles of operations

Given an atomic bid Bid_{ij}, combinatorial on operations, we have to ensure that:

- if an atomic operation t_{ijk} in bid Bid_{ij} is included in the solution sequence,
  - it must be included in the solution as many times as required by the multiplicity of t_{ijk} in the bid ($\mathcal{D}_{ij}(t_{ijk})$);
  - all the other atomic operations $t_{ijk'}$ within the same atomic bid (all the operations in $\mathcal{D}_{ij}$) must be included as many times as required by their multiplicities ($\mathcal{D}_{ij}(t_{ijk'})$) as well;
- the money that the bidder must pay (receive) is the price of the whole bid (p_{ij}).

Recall that bidder constraint 1.a) in Section 3.2 imposed that if a bidder submits an offer over a bundle of operations, all of them must be employed in the operation sequence. This is precisely what the first condition above captures.

We achieve this by introducing some auxiliary places and transitions. The example in Figure 4 represents the following bid:

$$Bid_{ij} = \text{BID}(1't_{ij1} + 3't_{ij2} + 2't_{ij3}, -20)$$

where $t_{ij1} = (2'p_1, 1'p_2 + 2'p_3)$, $t_{ij2} = (1'p_4, 1'p_6 + 1'p_7)$ and $t_{ij3} = (1'p_5, 1'p_8 + 1'p_9)$.

In general, in order to incorporate a bid over multiple operations we proceed as follows:

- for each bid Bid_{ij} we introduce an auxiliary transition t_{ij} (*bid transition*) and an auxiliary place c_{ij} (*bid place*).
- for each atomic operation t_{ijk} within bid Bid_{ij}, we add an auxiliary place c_{ijk} (c_{ij1}, c_{ij2}, and c_{ij3} in Figure 4), called *operation place*.

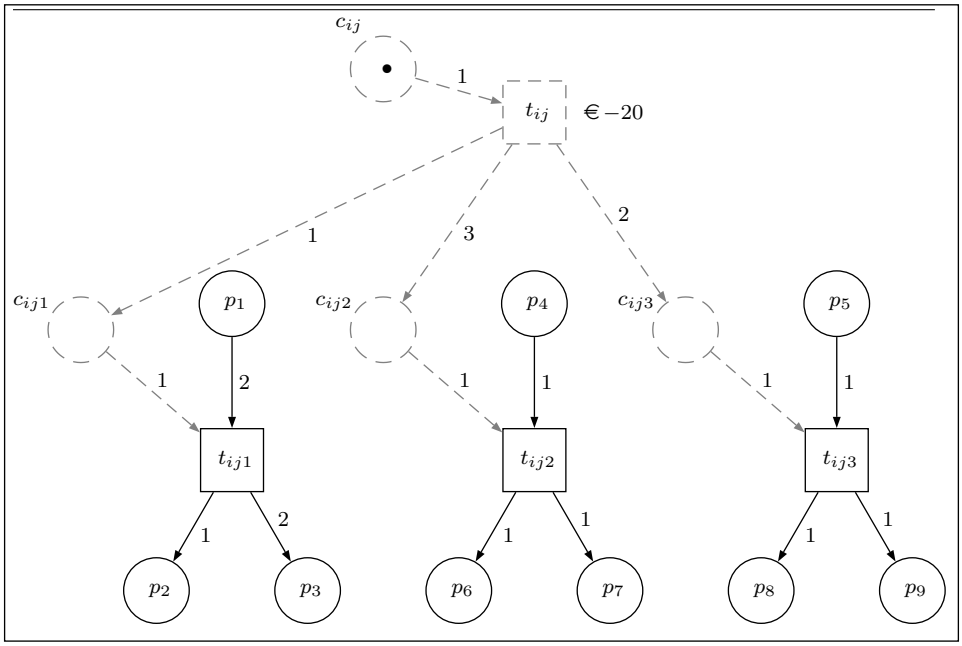

Fig. 4 Bids on bundles of operations.

– we attach the valuation p_{ij} of bid Bid_{ij} to the corresponding *bid transition* t_{ij}. In the example, we associate the bid value p_{ij}=−€20 to transition t_{ij}. Hence, whenever t_{ij} fires, the cost/value p_{ij} is added to the value of the firing sequence.

It is easy to check that the weighted net in Figure 4 allows to fire any subset of $\{t_{ij1}, t_{ij2}, t_{ij3}\}$ (depending on the tokens) with the corresponding multiplicities $(1, 3, 2)$. Notice also that firing at least one of the three transitions requires to previously fire transition t_{ij}, because this guarantees having the required tokens in the input places c_{ijk}. In this way, we guarantee that firing at least one of the transitions implies firing also t_{ij}, and therefore that the corresponding money is added to the overall cost/value.

Any legal firing sequence on the weighted net in Figure 4 guarantees that selecting at least one of the t_{ijk} implies also selecting t_{ij}. However, we need a further requirement: either none of the t_{ijk} fires, or all of them fire. If they all fire, they have to fire as many times as expressed by their multiplicities in the bids. In the figure, we have to enforce that if t_{ij} fires, then t_{ij1} fires once ($\mathcal{D}_{ij}(t_{ij1}) = 1$), t_{ij2} fires three times ($\mathcal{D}_{ij}(t_{ij2}) = 3$), and t_{ij3} fires twice ($\mathcal{D}_{ij}(t_{ij3}) = 2$).

The weighted net in Figure 4 cannot guarantee such property by itself. For instance, a firing sequence in which only transitions t_{ij1} and t_{ij2} fire (not t_{ij3}) is legal but does not comply with our all-or-nothing assumption. In order to enforce it, we simply impose some constraints on the final configuration of the net. Say that we impose that in the final configuration c_{ij1}, c_{ij2}, and c_{ij3} contain no tokens. More formally, the final marking should fulfil the constraints:

$$\mathcal{M}_d(c_{ij1}) = 0 \tag{29}$$

$$\mathcal{M}_d(c_{ij2}) = 0 \tag{30}$$

$$\mathcal{M}_d(c_{ij3}) = 0 \tag{31}$$

This implies that all the legal firing sequences leading to the final configuration $\mathcal{M}_d$ contain either none or the three transitions $t_{ij1}, t_{ij2}, t_{ij3}$ with multiplicities 1, 2, and 3 respectively.

Therefore, the semantics regarding the multiplicity of the operations offered in bid Bid_{ij} is completely captured by the weighted net provided. Furthermore, the weights of the arcs connecting each bid transitions t_{ij} with its operation places c_{ijk}, along with the constraints on the final marking, enforce that either none of the operations in $\mathcal{D}_{ij}$ is used, or all of them are used as many times as indicated by their multiplicities in $\mathcal{D}_{ij}$.

The following proposition formalises how a weighted net structure as constructed above can capture the semantics of an atomic bid.

Proposition 2 *Let Bid_{ij} be an atomic bid with value p_{ij} and its corresponding weighted structure (P, T, A, E, V) with initial marking $\mathcal{M}_0(c_{ij}) = 1$, and $\mathcal{M}_0(c_{ijk}) = 0$ for all operation places. If any final marking $\mathcal{M}_d$ is required to fulfil that $\mathcal{M}_d(c_{ijk}) = 0$ for all operation places, then any legal firing sequence fires either all or none of the atomic operations in the bid, being p_{ij} the value of the firing sequence in the first case and 0 otherwise.*

Proof Since $M_0(c_{ij}) = 1$, bid transition t_{ij} is enabled and thus can be fired. We distinguish two cases at this point. On the one hand, if t_{ij} does not fire, none of the atomic operations can fire, the constraints on the final marking hold and the value of not firing t_{ij} is 0. On the other hand, if t_{ij} fires, then each operation place c_{ijk} receives $\mathcal{D}_{ij}(t_{ijk})$ tokens from transition t_{ij}. Since each transition t_{ijk} requires a single token to be enabled, all atomic transitions are enabled. Since we have imposed that the final marking $\mathcal{M}_d$ leaves no tokens at the operation places, namely $\mathcal{M}_d(c_{ijk}) = 0\ \forall c_{ijk}$, all atomic transitions must fire. Therefore, both the bid transition t_{ij} along with all atomic transitions t_{ijk} compose the firing sequence. Since firing atomic transitions has no cost, the value of the firing sequence is $V(t_{ij}) = p_{ij}$, which is the value of bid Bid_{ij}. □

5.2.2 Expressing XOR of atomic bids

We have learnt how to represent an atomic bid on a weighted net. However, we still have to encode the XOR relationships among the atomic bids that come from each bidder to fully represent our bidding language. Consider the following bid:

$$\begin{gathered}\textsc{bid}(1't_{ij1} + 3't_{ij2} + 2't_{ij3}, -20)\\ XOR\\ \textsc{bid}(1't_{ij'1} + 1't_{ij'2}, -10)\end{gathered}$$

where $t_{ij1} = (2'p_1, 1'p_2 + 2'p_3)$, $t_{ij2} = (1'p_4, 1'p_6 + 1'p_7)$, $t_{ij3} = (1'p_5, 1'p_8 + 1'p_9)$, $t_{ij'1} = (3'p_5, 2'p_8 + 2'p_9)$, and $t_{ij'2} = (\{\}, 2'p_4 + 2'p_5)$.

Hereafter we refer to the two bids above submitted by some bidder i in XOR as to Bid_{ij} and $Bid_{ij'}$. Figure 5 depicts bids Bid_{ij} and $Bid_{ij'}$.

In order to incorporate the semantics of the XOR operator into a weighted net, we introduce a new place, labelled with p_i^{XOR}, called an *XOR place*. Notice that an *XOR place* replaces *bid places*. In particular, in figure 5 it replaces *bid places* c_{ij} and $c_{ij'}$ corresponding to bids Bid_{ij} and $Bid_{ij'}$. An *XOR place* has no input arcs and is

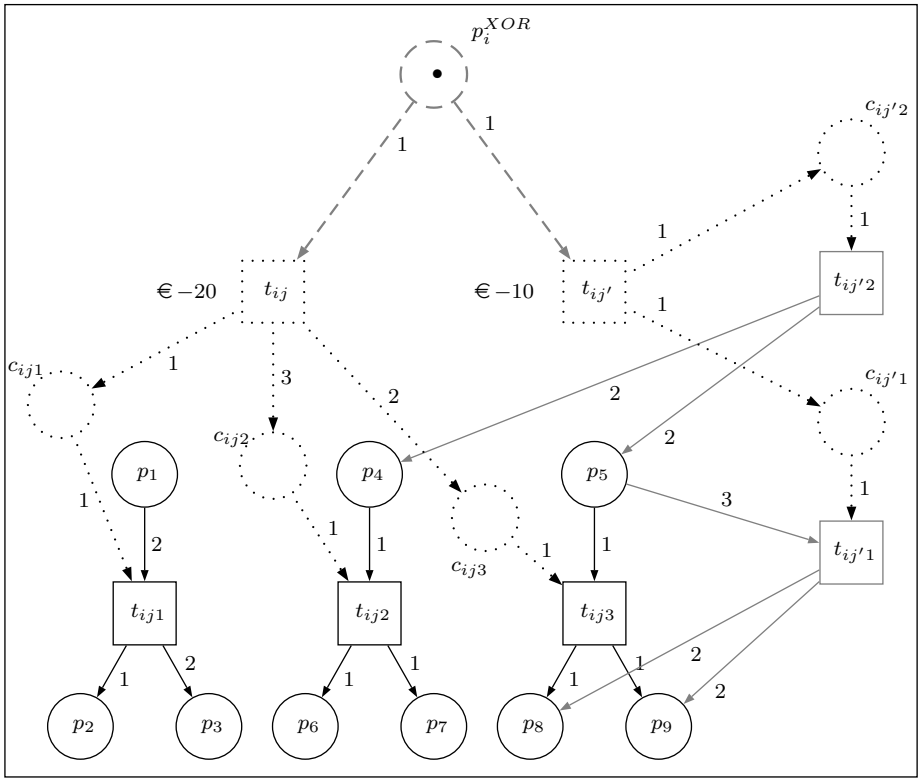

Fig. 5 XOR of atomic bids.

enforced to contain a single token by the initial marking. Hence, the resulting weighted net topology enforces that at most one of the atomic bids in XOR is selected. In figure 5 either t_{ij} or $t_{ij'}$ can fire, but not both. When either of them fires, it consumes the unique token in p_i^{XOR}, thus preventing the firing of the other one. This corresponds to selecting at most one bid out of bids Bid_{ij} and $Bid_{ij'}$. This reasoning naturally applies to the general case of m bids in XOR. Moreover, say that, for instance, we choose to fire t_{ij}. Then all atomic operations in the bid (namely $t_{ij1}, t_{ij2}, t_{ij3}$) must fire as stated by proposition 2. In other words, an *XOR place* acts as an exclusive bid place for all the atomic bids involved in an XOR bid.

5.3 Connecting the mixed auction WDP and MAXSEQ: the Mixed Auction Net

In this section we formalize what was explained informally in the previous section: how to build a weighted net that encodes all the bids received by an auctioneer. We shall call such weighted net the *Mixed Auction Net*.

Definition 13 Given a finite set of bids B in the *XOR* bidding language and a multiset $\mathcal{U}_{in}$ over a set of goods G, its *Mixed Auction Net* is the weighted net computed by function **ConstructMixedAuctionNet**(B,$\mathcal{U}_{in}$) in Algorithm 1. □

Algorithm 1 details how to build a mixed auction net from some set B of XOR bids. Line 1 creates a place for each good at auction and initialises the remaining variables to empty sets. We call P_G the set that contains the places that represent goods. Lines 2-4 establish that the initial marking for places that represent goods is set according to the number of units that $\mathcal{U}_{in}$ specifies for each good. Line 7 adds a new place (p_i^{XOR}) for each XOR bid and adds a single token (unit) to the initial marking (so that each XOR bid can be only selected once). The P_{XOR} set contains all XOR places. Line 10 adds a new transition (t_{ij}) for each atomic bid and sets its value to p_{ij} (the value specified by the bid), whereas line 11 links each newly created transition with the place representing its XOR bid. Line 13 adds a *control* place c_{ijk} to control each operation and line 14 links it as an output to the transition representing its combinatorial bid (t_{ij}), setting the number of tokens to output to the cardinality of operation t_{ijk} in the combinatorial bid (namely to $\mathcal{D}_{ij}(t_{ijk})$). The P_C set contains all the control places. Line 16 adds a new transition t_{ijk} (with value zero) for each atomic bid, which line 17 links to its control place c_{ijk} so that each time t_{ijk} fires, it consumes a token from c_{ijk}. We call the set of all transitions representing operations T_{OP}. Finally, lines 18-20 and 21-23 link each transition t_{ijk} representing an operation with its input and output goods respectively.

The introduction of the mixed auction net allows us to define the mixed auction WDP as a MAXSEQ problem.

Theorem 4 *Given a mixed auction with a multiset of available goods $\mathcal{U}_{in}$, a multiset of required goods $\mathcal{U}_{out}$, and a set of bids B in the XOR language over the goods in G, solving the WDP amounts to solving the MAXSEQ problem defined on the* Mixed

Algorithm 1 ConstructMixedAuctionNet(B,$\mathcal{U}_{in}$)

1: $P \leftarrow \{p_g \mid g \in G\}$; $T \leftarrow \emptyset$; $A \leftarrow \emptyset$; $\mathcal{M}_0 \leftarrow \emptyset$;
2: **for** g in G **do**
3: $\quad \mathcal{M}_0 \leftarrow \mathcal{M}_0 + \{\mathcal{U}_{in}(p_g)$ copies of $p_g\}$; /* Establish initial marking for goods */
4: **end for**
5: **for** Bid_i in B **do**
6: $\quad$ /* Add a place for each XOR bid and set its initial marking to contain one token */
7: $\quad P \leftarrow P \cup \{p_i^{XOR}\}$; $\mathcal{M}_0 \leftarrow \mathcal{M}_0 + \{p_i^{XOR}\}$;
8: $\quad$ **for** Bid_{ij} in Bid_i **do**
9: $\quad\quad$ /* Add a new transition for each atomic bid, set its value to the bid's value */
10: $\quad\quad T \leftarrow T \cup \{t_{ij}\}$; $V(t_{ij}) = p_{ij}$;
11: $\quad\quad A \leftarrow A \cup \{(p_i^{XOR}, t_{ij})\}$; $E(p_i^{XOR}, t_{ij}) = 1$; /* and link it to its XOR bid */
12: $\quad\quad$ **for** Bid_{ijk} in Bid_{ij} **do**
13: $\quad\quad\quad P \leftarrow P \cup \{c_{ijk}\}$; /* Add a new place for each operation */
14: $\quad\quad\quad A \leftarrow A \cup \{(t_{ij}, c_{ijk})\}$; $E(t_{ij}, c_{ijk}) = \mathcal{D}_{ij}(t_{ijk})$; /* and link it to its combin. bid */
15: $\quad\quad\quad$ /* Add a new transition for each operation, set its value to zero */
16: $\quad\quad\quad T \leftarrow T \cup \{t_{ijk}\}$; $V(t_{ij}) = 0$;
17: $\quad\quad\quad A \leftarrow A \cup \{(c_{ijk}, t_{ijk})\}$; $E(c_{ijk}, t_{ijk}) = 1$; /* link it to its control place, */
18: $\quad\quad\quad$ **for** g in $\mathcal{I}_{ijk}$ **do**
19: $\quad\quad\quad\quad A \leftarrow A \cup \{(p_g, t_{ijk})\}$; $E(p_g, t_{ijk}) = \mathcal{I}_{ijk}(p_g)$; /* link it to its inputs */
20: $\quad\quad\quad$ **end for**
21: $\quad\quad\quad$ **for** g in $\mathcal{O}_{ijk}$ **do**
22: $\quad\quad\quad\quad A \leftarrow A \cup \{(t_{ijk}, p_g)\}$; $E(t_{ijk}, p_g) = \mathcal{O}_{ijk}(p_g)$; /* and link it to its outputs */
23: $\quad\quad\quad$ **end for**
24: $\quad\quad$ **end for**
25: $\quad$ **end for**
26: **end for**
27: **return** $(P, T, A, E, \mathcal{M}_0, V)$;

Auction Net $S = (P, T, A, E, \mathcal{M}_0, V)$ *with destination marking* $\mathcal{M}_d$ *that fulfils the following constraints*

$$\mathcal{M}_d(p) \geq \mathcal{U}_{out}(g) \qquad p \in P_G \tag{32}$$

$$\mathcal{M}_d(p) = 0 \qquad p \in P_C \tag{33}$$

$$\mathcal{M}_d(p) \geq 0 \qquad p \in P_{XOR} \tag{34}$$

Proof The proof is long and needs a bit more notation than introduced in this paper. We only report about the way the two solutions can be mapped to each other. For the complete proof the reader can consult [25].
⇒) First, we prove that a solution to MAXSEQ can be transformed into a solution to the WDP. Note that each solution to MAXSEQ is a sequence of transitions, some of them representing atomic bids and some others representing operations. A solution to the mixed auction WDP is composed by two sets of binary variables x^m_{ijk} (that takes on 1 if t_{ijk} is at position m in the solution sequence) and x_{ij} (that takes on 1 if bid Bid_{ij} is accepted). We construct the mixed auction WDP solution by removing from the MAXSEQ solution all the transitions that represent atomic bids. In the resulting sequence, transitions only represent operations. Hence, we can set to 1 those variables x^m_{ijk} such that operation t_{ijk} appears at position m. Regarding the transitions in the MAXSEQ solution that represent atomic bids, we can set x_{ij} to 1 if t_{ij} appears in the MAXSEQ solution. It is then straightforward (although tedious and beyond the scope of this paper) to show that this assignment of values to x^m_{ijk} and x_{ij} fulfils the conditions in equations (1)-(6) corresponding to the Direct Integer Program that solves the MMUCA WDP.
⇐) We prove the converse as well. Given a solution $\{x^m_{ijk}, x_{ij}\}$ to the mixed auction WDP, it can be transformed into a solution to the MAXSEQ problem posed in the theorem. In this case we construct a sequence of transitions by placing a transition t_{ij} (representing an atomic bid) in the sequence for each bid Bid_{ij} such that x_{ij} is 1. We can place these transitions at any order at the beginning of the sequence. We complete the remainder of the sequence with transitions that represent operations following the order specified by x^m_{ijk}. More precisely, if k atomic bids are selected for the solution (k variables x_{ij} taking on 1), then for every variable x^m_{ijk} with value 1 we can place the transition representing operation t_{ijk} at position $k + m$ of the transition sequence. It is also tedious but simple to show that such sequence fulfils the conditions to be a solution to the MAXSEQ problem. □

Notice that it is straightforward to generalise the previous result for the OR language by incorporating appropriate changes to the weighted net. Thus, we should only represent all the bids as in Figure 4, omitting the XOR places.

Now we can start benefitting from Theorem 4 by providing (under acyclicity) a better mapping of the mixed auction WDP to ILP than the one provided in Section 3.3.

Corollary 2 *Whenever a* mixed auction net *is acyclic, the mixed auction WDP can be reduced to ILP using* $|B| + |T|$ *variables, where* $|B|$ *stands for the number of atomic bids received and* $|T|$ *stands for the number of different operations appearing in the bids.*

Proof Follows directly from the fact that the mixed auction WDP can be reduced to MAXSEQ and that whenever the weighted net is acyclic MAXSEQ can be reduced to ILP using as many variables as transitions. □

Recall that the number of variables needed for the mapping in Section 3.3 is $O(\delta \cdot |T|)$, i.e. it is quadratic in $|T|$, and possibly even significantly larger than that, namely when transformations appear with high multiplicity. The acyclicity constraint excludes some interesting cases such as those where a good is borrowed and returned after the execution of the operation. However, it covers some common topologies such as the assembly of a good from its components or its dissasembly into its parts (though not both simultaneously).

Our aim in this section is showing that, from the firing sequence associated with a particular MAXSEQ problem on the *Mixed Auction Net*, we can derive an optimal solution sequence to the corresponding mixed auction WDP.

5.4 Advantages of the mapping to MAXSEQ

It is time to highlight the advantages brought about by mapping the mixed auction WDP to the MAXSEQ problem over weighted nets. In particular, the mapping allows to import all the Petri net tools and properties presented in the literature to analyze structural and behavioral properties of the supply chain resulting from a mixed auction. Some examples of application are listed below:

1. One can very efficiently solve the underlying ILP when the supply chain is acyclic. This benefit comes from exploiting an important nets analysis tool, the state equation.
2. One may be interested in maintaining under a certain threshold the level of resources present in each place (for instance, because of inventory capacity constraints). In order to guarantee that resources do not exceed some threshold(s) amounts to investigating the so-called boundedness property [7], a well-known behavioural property of nets.
3. Due to the very appealing and intuitive weighted net graphical representation, we can compactly encode and visualize the search space associated with the mixed auction WDP. This stems from the the fact that the semantics of transitions on nets naturally accommodates the representation of operations.
4. Once a solution sequence to the mixed auction WDP is obtained, one can visualize it by means of a token game showing the evolution of the supply chain at any step of the operation sequence.
5. One can graphically visualize the mixed auction WDP problem. This provides a very helpful guidance to obtaine insights about the problem. For instance, by visualizing the mixed auction WDP by means of a weighted net, one can incorporate new bidding language constructs with a minimum effort. For instance, consider the following example.

   Example 2 We explained that switching to the *OR* language instead of the *XOR* bidding language is as simple as removing the *XOR* place from Figure 5, as shown in Figure 4. However, there is another widely employed bidding language that is very compact and human-readable. Is is called the *XOR-of-OR* bidding language (refer to Section 3.1). When employing a XOR-of-OR bidding language, any XOR combination of OR combinations of atomic bids can be selected. For instance, the following bid:

$$((a,1)\ OR\ (a,1)\ OR\ (a,1)\ OR\ (a,1)\ OR\ (a,1))\ XOR\ (b,2) \tag{35}$$

means that an auctioneer can select from 0 to 5 copies of atomic bid $(a, 1)$ or one copy of atomic bid $(b, 2)$, but not both things at the same time. In Figure 6, we graphically show how to incorporate the *XOR-of-OR* bidding language by depicting the following bid:

$$(\text{BID}(1't_{ij1} + 3't_{ij2} + 2't_{ij3}, -20)\ \ OR \tag{36}$$

$$\text{BID}(1't_{ij'1} + 1't_{ij'2}, -10))\ \ XOR \tag{37}$$

$$\text{BID}(1't_{ij''1}, -2) \tag{38}$$

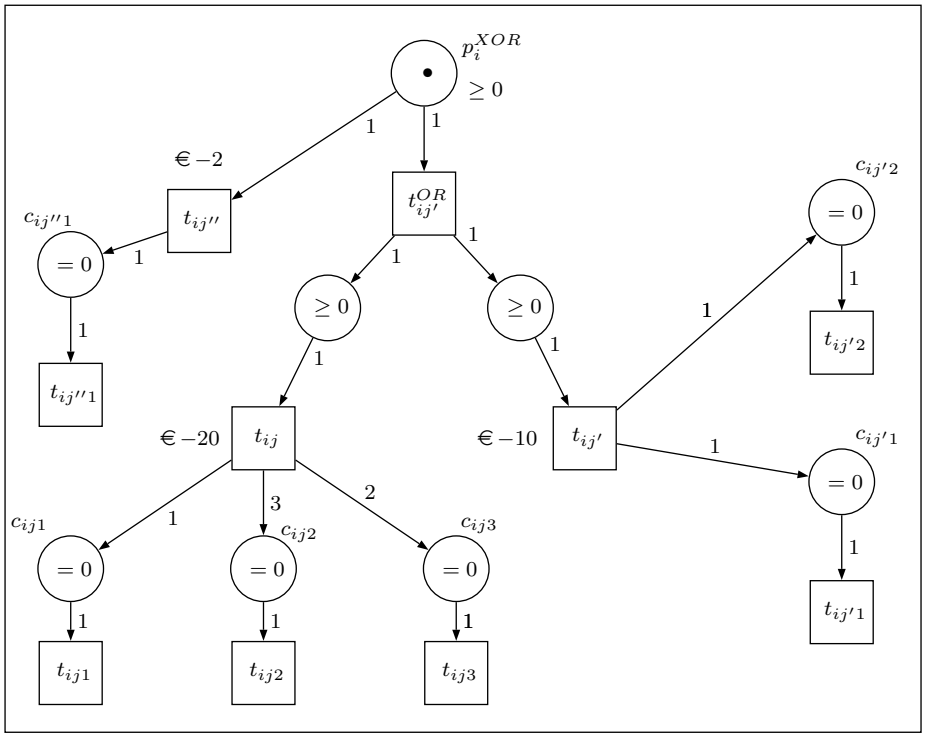

Fig. 6 XOR-of-OR of atomic bids

The single token in place p_i^{XOR} allows either to fire transition $t_{ij''}$ or (exclusively) transition $t_{ij'}^{OR}$. If transition $t_{ij''}$ fires, then the auctioneer would be using the bid $\text{BID}(1't_{ij''1}, -2)$. Otherwise, the firing of transition $t_{ij'}^{OR}$ would place exactly one token in each one of its output places. That would enable both transitions t_{ij} and $t_{ij'}$, and therefore the auctioneer could select firing either: (i) t_{ij} (namely accepting $\text{BID}(1't_{ij1} + 3't_{ij2} + 2't_{ij3}, -20)$); (ii) $t_{ij'}$ (namely accepting $\text{BID}(1't_{ij'1} + 1't_{ij'2}, -10)$)); (iii) both. All these combinations would fulfil the final constraints represented by the inequalities at the output places of transition $t_{ij'}^{OR}$. □

6 Conclusions and future work

Mixed auctions can potentially be employed for the automated assembly of supply chains of agents. However, in order for mixed auctions to be effectively applied to SCF, we must ensure computational feasibility while preserving optimality. In this paper we have tried to make headway along this direction.

Firstly we discussed the notions of bidding language and winner determination for mixed auctions following [4]. Integer programming allows to solve the mixed auction WDP on *any* supply chain network topology. However, it has the disadvantage to be computationally expensive. In fact, such a an ILP formulation requires a number of decision variables that grows quadratically with the number of operations mentioned in the bids. This computational cost motivates the need for efficient solvers that support the practical application of mixed auctions.

Contributions on computationally efficient WDP solvers for different auction types (namely, [22] for CAs and [29] for multi-attribute double auctions) agree on and defend that a careful, formal analysis of the structure of the WDP can provide guidance for developing efficient solvers. Along this line we have introduced a graphical formalism that allows to compactly represent both the search space and the solutions associated with the mixed auction WDP. To attain this goal we have extended Place Transition Nets to provide the so-called weighted nets, we have defined a new optimisation problem over weighted nets, the so-called Constrained Maximum Weight Occurrence Sequence Problem (MAXSEQ), and we have mapped the mixed auction WDP to a MAXSEQ. Notice that the validity of the mapping from the mixed auction WDP to weighted nets is not restricted to bids in the XOR language, but in fact it can easily cope with other languages. For instance, as discussed in Section 5.4, the extension to the OR and the OR-of-XOR language is trivial.

A major benefit of our graphical formalism is that it allows to formally analyse the structural and behavioural properties of the mixed auction WDP. In this work, as a first result of our structural analysis, we demonstrate how to dramatically reduce the number of decision variables (from quadratic to linear with respect to the number of operations) for a broad class of mixed auctions WDPs (in particular, when the weighted nets underlying the mixed auction WDP is acyclic).

Notice that although our approach is limited to a class of mixed auction WDPs, we know from the literature that it is possible to increase the classes of Petri nets for which the state equation represents the whole reachability set. As an example, one may add linear side constraints to the state equation [30]. In general, we would like to broaden the class of mixed auction WDPs we can efficiently solve. Thus, our future aim will be to devise the most efficient solver for each class of mixed auction WDP.

This work opens several paths to future research. The most interesting extension we envisage to mixed auctions concerns the incorporation of time and uncertainty. On the one hand, there is the need to express the release time and duration of operations, as well as their inclusion in the WDP. Hence, an auctioneer would be able to fix deadlines to have his production process completed. Moreover, the participants in the supply chain would be able to synchronise their operations by fulfilling not only the producer/consumer relationships, but also time constraints. On the other hand, an auctioneer might be interested in assigning a probability of success to each operation to obtain more robust supply chains that prevent failures and shortcomings. In both cases, we would require to extend our current model of weighted net along with the mapping to the mixed auction WDP.

Acknowledgements

We would like to thank the anonymous reviewers for their extensive and helpful comments. This work was funded by the Jose Castillejo programme (JC2008-00337), IEA (TIN2006-15662-C02-01), OK (IST-4-027253-STP), eREP(EC-FP6-CIT5-28575) and Agreement Technologies (CONSOLIDER CSD2007-0022, INGENIO 2010).

References

1. P. Cramton, Y. Shoham, R. Steinberg (Eds.), Combinatorial Auctions, MIT Press, 2006.
2. W. E. Walsh, M. P. Wellman, F. Ygge, Combinatorial auctions for supply chain formation, in: EC '00: Proceedings of the 2nd ACM conference on Electronic commerce, ACM, New York, NY, USA, 2000, pp. 260–269.
3. W. Walsh, M. Wellman, Decentralized supply chain formation: A market protocol and competitive equilibrium analysis, Journal of Artificial Intelligence Research 19 (2003) 513–567.
4. J. Cerquides, U. Endriss, A. Giovannucci, J. A. Rodriguez-Aguilar, Bidding languages and winner determination for mixed multi-unit combinatorial auctions, in: Proc. of the 20th Intl. Joint Conferences on Artif. Intelligence (IJCAI), Hyderabad, India, 2007, pp. 1221–1226.
5. M. Vinyals, A. Giovannucci, J. Cerquides, P. Meseguer, J. A. Rodriguez-Aguilar, A test suite for the evaluation of mixed multi-unit combinatorial auctions, Journal of Algorithms 63 (1-3) (2008) 130–150.
6. A. Giovannucci, J. A. Rodriguez-Aguilar, J. Cerquides, U. Endriss, Winner determination for mixed multi-unit combinatorial auctions via petri nets, in: AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems, ACM, New York, NY, USA, 2007, pp. 710–717.
7. T. Murata, Petri nets: Properties, analysis and applications, in: Proceedings of the IEEE, Vol. 77, 1989, pp. 541–580.
8. T. Sandholm, S. Suri, A. Gilpin, D. Levine, Winner determination in combinatorial auction generalizations, in: AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems, ACM Press, Bologna, Italy, 2002, pp. 69–76.
9. C. Caplice, Y. Sheffi, Combinatorial Auctions for Truckload Transportation, MIT Press, 2006, Ch. 21. Combinatorial Auctions.
10. E. Cantillon, M. Pesendorfer, Auctioning Bus Routes: The London Experience, MIT Press, 2006, Ch. 22. Combinatorial Auctions.
11. M. Bichler, A. Davenport, G. Hohner, J. Kalagnanam, Industrial Procurement Auctions, MIT Press, 2006, Ch. 23. Combinatorial Auctions.
12. M. O. Ball, G. L. Donohue, K. Hoffman, Auctions for the Safe, Efficient, and Equitable Allocation of Airspace System Resources, MIT Press, 2006, Ch. 20. Combinatorial Auctions.
13. A. Pekec, M. H. Rothkopf, Combinatorial auction design, Manage. Sci. 49 (11) (2003) 1485–1503.
14. V. Krishna, Auction Theory, Academic Press, 2002.
15. P. Milgrom, Putting Auction Theory to Work, Cambridge University Press, 2004.
16. N. Nisan, Bidding Languages for Combinatorial Auctions, MIT Press, 2006, Ch. 9. Combinatorial Auctions.
17. M. H. Rothkopf, A. Pekec, R. M. Harstad, Computationally manageable combinational auctions, Management Science 44 (8) (1998) 1131–1147.
18. A. Andersson, M. Tenhunen, F. Ygge, Integer programming for combinatorial auction winner determination, in: Fourth International Conference on Multiagent Systems (ICMAS 2000), Boston, MA, 2000, pp. 39–46.
19. T. Sandholm, Algorithm for optimal winner determination in combinatorial auctions, Artificial Intelligence 135 (1-2) (2002) 1–54.
20. Y. Fujishima, K. Leyton-Brown, Y. Shoham, Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches., in: Proceeding of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99), 1999, pp. 548–553.

21. K. Leyton-Brown, Y. Shoham, M. Tennenholtz, An algorithm for multi-unit combinatorial auctions, in: Proceedings of the American Association for Artificial Intelligence Conference (AAAI), 2000, pp. 56–61.
22. D. Lehmann, R. Müller, T. Sandholm, The winner determination problem, MIT Press, 2006, Ch. 12. Combinatorial Auctions.
23. R. Müller, Tractable Cases of the Winner Determination Problem, MIT Press, 2006, Ch. 13. Combinatorial Auctions.
24. T. Sandholm, Optimal Winner Determination Algorithms, MIT Press, 2006, Ch. 14. Combinatorial Auctions.
25. A. Giovannucci, Computationally Manageable Combinatorial Auctions for Supply Chain Automation, Ph.D. thesis, Universitat Autonoma de Barcelona. Departamento de ciencias de la Computacion. `http://specs.upf.edu/files/u20/Monografia.pdf`. (2008).
26. C. Petri, Kommunikation mit Automaten.Univ. Bonn, Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962., Tech. rep., English translation: RADC-TR-65-377, Griffiths Air Base, New York (1966).
27. R. Lipton, The reachability problem requires exponential space, Tech. Rep. 62, Yale University (1976).
28. A. Tarek, N. Lopez-Benitez, Optimal legal firing sequence of Petri nets using linear programming, Optimization and Engineering 5 (1) (2004) 25–43.
29. Y. Engel, M. P. Wellman, K. M. Lochner, Bid expressiveness and clearing algorithms in multiattribute double auctions, in: EC '06: Proceedings of the 7th ACM conference on Electronic commerce, ACM Press, New York, NY, USA, 2006, pp. 110–119.
30. J. Esparza, S. Melzer, Verification of safety properties using integer programming: Beyond the state equation, Formal Methods in System Design 16 (2004) 159–189.