Artificial Intelligence 196 (2013) 1–25

Contents lists available at SciVerse ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

The complexity of mixed multi-unit combinatorial auctions: Tractability under structural and qualitative restrictions

Valeria Fionda [a], Gianluigi Greco [b,*]

[a] *Faculty of Computer Science, Free University of Bozen-Bolzano, Italy*
[b] *Department of Mathematics, University of Calabria, Italy*

ARTICLE INFO

Article history:
Received 7 November 2011
Received in revised form 17 December 2012
Accepted 23 December 2012
Available online 3 January 2013

Keywords:
Mixed multi-unit combinatorial auctions
Computational complexity
Structural decomposition methods

ABSTRACT

Mixed multi-unit combinatorial auctions (MMUCAs) are extensions of classical combinatorial auctions (CAs) where bidders trade transformations of goods rather than just sets of goods. Solving MMUCAs, i.e., determining the sequences of bids to be accepted by the auctioneer, is computationally intractable in general. However, differently from classical combinatorial auctions, little was known about whether polynomial-time solvable classes of MMUCAs can be singled out on the basis of their characteristics. The paper fills this gap, by studying the computational complexity of MMUCA instances under structural and qualitative restrictions, which characterize interactions among bidders and types of bids involved in the various transformations, respectively.

1. Introduction

Mixed multi-unit combinatorial auctions (MMUCAs) are extensions of classical combinatorial auctions (CAs) where participants are allowed to bid not only on bundles of goods to buy, but also on bundles of goods to sell and of transformations of goods [1].

These mechanisms are particularly useful in the context of automatizing *supply chain formation*, where production processes often emerge as the result of complex interactions among producers and consumers [2]. Indeed, in these contexts, the auctioneer wants to obtain certain products based on the goods she initially owns, by exploiting a production process possibly involving further goods to be acquired from suppliers or to be obtained via transformations operating on the goods currently available to her.

Example 1.1. Consider the supply chain associated with the production of bicycles, which is illustrated in Fig. 1 according to an intuitive graphical notation where goods are represented as ovals, transformations as boxes, and where arrows indicate inputs and outputs of the various transformations. Assume that the auctioneer is presented with 6 different bids over the singleton sets of transformations $\{t_1\}, \{t_2\}, \ldots, \{t_6\}$, whose associated prices are then reported in the boxes as well.

The assembly of a bicycle from its constituents (i.e., frame, brakes, drive train, front wheel, back wheel, seat, and handlebars) is thus offered to the auctioneer through the bid over $\{t_6\}$, which is sold for \$10. However, the auctioneer owns only a subset of such constituents (i.e., frame and brakes), plus two goods (i.e., chain and chainring) that are not immediately exploitable by t_6. The auctioneer has therefore to ask suppliers for the missing goods (i.e., front wheel, back wheel, seat, and handlebars), which are made available trough the bids over $\{t_3\}$, $\{t_4\}$, and $\{t_5\}$, for \$10, \$17, and \$8, respectively. Note that t_4 incidentally produces a dynamo, which is not part of the bicycle the auctioneer is willing to produce. Finally, note that the auctioneer

* Corresponding author.
E-mail addresses: fionda@inf.unibz.it (V. Fionda), ggreco@mat.unical.it (G. Greco).

0004-3702/\$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.artint.2012.12.002

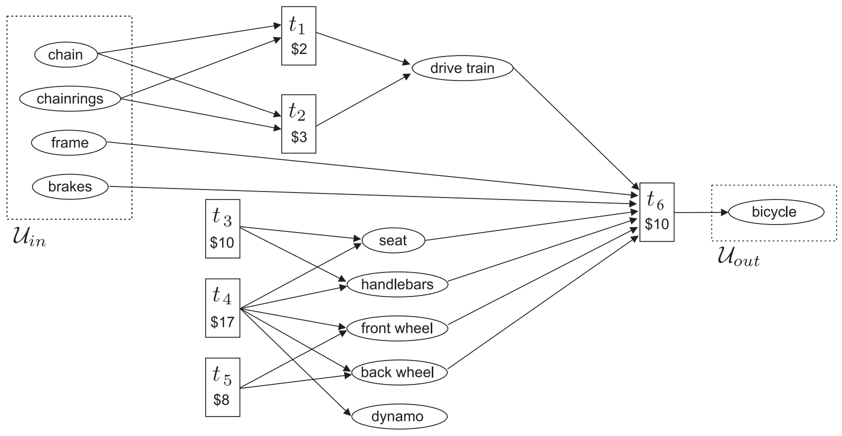

Fig. 1. Mixed multi-unit combinatorial auction in Example 1.1.

needs to accept one further bid (either over $\{t_1\}$ for \$2, or over $\{t_2\}$ for \$3) to assemble the chain and the chainrings into the drive train, in order for the latter to be taken as input by t_6. ◁

A solution to a MMUCA instance is, roughly, any set of bids whose transformations can be arranged in a sequence allowing the auctioneer to produce the desired goods. Among all possible solutions, in the WINNER-DETERMINATION problem we are interested in singling out those having the minimum total cost or, equivalently, guaranteeing the maximum possible revenue. For instance, it is easily seen that the minimum production cost in Example 1.1 is \$29, which is witnessed by the sequence of transformations t_1, t_4, t_6.

The WINNER-DETERMINATION problem for MMUCA instances has intensively been studied in recent years, by extending to this novel setting several results originally conceived for classical CAs. In particular, languages have been defined and analyzed which allow bidders to compose (atomic) bids in a natural and intuitive way [1], and motivated by their intractability (formally, NP-hardness), solution approaches have been proposed (see, e.g., [3]) that well-behave on realistic scenarios [4]. Differently from classical CAs, however, little was known about whether polynomial-time solvable classes of MMUCAs can be singled out based on the structural and topological properties of the instances at hand. As a matter of fact, by focusing on the kinds of interactions among bidders that are likely to occur in practice, classes of instances over which WINNER-DETERMINATION is tractable—called “islands of tractability” in the literature—have been identified for classical CAs (such as *structured item graphs* [5] or *bounded hypertree-width dual hypergraphs* [6]). However, none of these results had a counterpart in the case of MMUCAs.

The aim of this paper is precisely to fill this gap, by depicting a clear and complete picture of the frontier of tractability for MMUCA instances, with respect to both *qualitative* and *structural* parameters. In particular, note that the existence of a solution is not guaranteed in the case of MMUCAs. For instance, in Example 1.1, if the auctioneer does not initially own the frame, then no solution exists at all. Therefore, checking for the feasibility of the production process is an important and peculiar source of complexity for MMUCA instances and, accordingly, attention will be focused not only on the WINNER-DETERMINATION but also on the FEASIBILITY problem of deciding whether a given instance admits a solution at all (no matter of its cost).

Our contribution can be summarized as follows:

(1) We chart the tractability frontier of the FEASIBILITY problem for MMUCAs under *qualitative* restrictions, i.e., under restrictions characterizing the types of bids in terms of the variety and quantity of goods involved in the various transformations. The analysis is carried out over three different bidding languages:
- *Atomic bids*, where each bid is just defined over one set of transformations—this is the building block of the following two languages;
- OR-*language*, where bidders submit sets of atomic bids and accept to implement any combination of them for the sum of their prizes; and
- XOR-*language*, where each bidder accepts to implement at most one atomic bid from the set of hers submitted atomic bids.

In particular, for the above bidding languages, we analyze the scenario where each underlying atomic bid is defined over a set containing one transformation only (as in Example 1.1), as well as the more general case where each atomic bid is defined over an arbitrary set of transformations.

(2) We study the complexity of FEASIBILITY under *structural* restrictions of the networks originating from bidder interactions, motivated by the fact that many NP-hard problems in different application areas are known to be efficiently solvable when restricted to instances that can be modeled via (nearly)acyclic instances. Surprisingly, bad news emerged from our investigation. Indeed, we show that Feasibility is hard on (nearly)acyclic instances too, and even for atomic bids only. In particular, this is the case for two natural ways of encoding bidder interactions, namely, for:

- *transformations graphs*, where nodes are in one-to-one correspondence with transformations and an edge indicates that one transformation produces a good required by the other, and for
- *goods graphs*, where nodes are in one-to-one correspondence with goods and an edge indicates the possibility of transforming a good into another.

(3) We study the complexity of the Winner-Determination problem on MMUCA instances, in order to single out tractable classes of MMUCAs extending those defined in [5,6] for CAs. To this end, we propose a notion of *intricacy* of an instance and we define a hypergraph encoding for bidders interactions. The two concepts are designed to evidence the sources of qualitative and structural intractability, respectively, which emerged in our analysis of the complexity of the Feasibility problem. In fact, we show that on classes of instances with “small intricacy” and whose associated hypergraphs are (nearly)acyclic, Winner-Determination can be solved in polynomial time.

Note that the analysis we carry out in this paper extends some preliminary results on the complexity of MMUCAs we discussed in [7]. There, we focused in fact on atomic bids, in a setting where each bid is defined over a set containing one transformation, and where the free disposal assumption has been considered only. Moreover, the hypergraph-based approach to encode interactions among bidders and the associated results are entirely novel contributions of this paper.

Organization The rest of the paper is organized as follows. Section 2 reports a few preliminaries on MMUCAs. The complexity of Feasibility under qualitative and structural restrictions is discussed in Section 3 and Section 4, respectively. Tractability islands for the Winner-Determination problem are isolated in Section 5, and conclusions are drawn in Section 6.

2. Mixed multi-unit combinatorial auctions

In this section, we recall some basic notions about MMUCAs, and we illustrate relevant results known in the literature on this formalism.

2.1. Formal framework

Let G be a set of types of goods. Any function $\mathcal{W}: G \to \mathbb{N}$ will be hereafter equivalently viewed as a multi-set over G containing $\mathcal{W}(g)$ repetitions of each good $g \in G$. As an example, $\mathcal{W} = \{\}$ denotes the function $\mathcal{W}: G \to \mathbb{N}$ such that $\mathcal{W}(g) = 0$, for each $g \in G$. Moreover, in order to manipulate such multi-sets, we shall use standard operators, such as ‘$\cap$’, ‘$\cup$’, ‘$\setminus$’, ‘$\subseteq$’, and ‘$=$’, under their usual semantics. For instance, $\{g, g\} \setminus \{g\} = \{g\}$ and $\{g, g\} \cap \{g\} = \{g\}$.

A *transformation* over G is a tuple $\langle \mathcal{I}, \mathcal{O} \rangle$ where $\mathcal{I}: G \to \mathbb{N}$ (resp., $\mathcal{O}: G \to \mathbb{N}$) is a function mapping each good $g \in G$ to the quantity required (resp., produced) for the transformation to take place (resp., as a result of the transformation). If $\mathcal{I} = \{\}$ (resp., $\mathcal{O} = \{\}$), then the transformation is just meant at offering (resp., requesting) some goods.

An *atomic bid* over G is a triple $\mathbf{b} = \langle \mathcal{B}, p, \textsc{type} \rangle$, where $\mathcal{B}$ is a multi-set of transformations over G and where $p \in \mathbb{R}$ is the payment the bidder is willing to make in return for being allocated *all* transformations in $\mathcal{B}$ or *any subset* $\mathcal{B}' \subseteq \mathcal{B}$, depending on whether $\textsc{type} = \textsc{full}$ or $\textsc{type} = \textsc{partial}$, respectively. Note that if $p < 0$, then the auctioneer must actually pay $-p$ to the bidder in order for her to implement the transformations.

A *mixed multi-unit combinatorial auction* instance *over atomic bids* is a tuple $\mathcal{A} = \langle G, \mathfrak{B}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$, where G is a set of goods, $\mathfrak{B}$ is a multi-set of atomic bids over G, and $\mathcal{U}_{in}: G \to \mathbb{N}$ (resp., $\mathcal{U}_{out}: G \to \mathbb{N}$) is a function denoting the quantities of goods the auctioneer holds to begin with (resp., expects to end up with).

Definition 2.1 *(Solutions).* Let $\mathfrak{B}' \subseteq \mathfrak{B}$ be a multi-set of atomic bids over a set G of types of goods. We say that $\mathfrak{B}'$ is a *feasible outcome* for $\mathcal{A} = \langle G, \mathfrak{B}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$ if there is a sequence $\sigma = \langle \mathcal{I}_1, \mathcal{O}_1 \rangle, \ldots, \langle \mathcal{I}_k, \mathcal{O}_k \rangle$ of transformations such that:

(1) $\{\langle \mathcal{I}_1, O_1 \rangle, \ldots, \langle \mathcal{I}_k, \mathcal{O}_k \rangle\} \supseteq \bigcup_{\langle \mathcal{B}, p, \text{FULL} \rangle \in \mathfrak{B}'} \mathcal{B}$;
(2) $\{\langle \mathcal{I}_1, O_1 \rangle, \ldots, \langle \mathcal{I}_k, \mathcal{O}_k \rangle\} \setminus \bigcup_{\langle \mathcal{B}, p, \text{FULL} \rangle \in \mathfrak{B}'} \mathcal{B} \subseteq \bigcup_{\langle \mathcal{B}, p, \text{PARTIAL} \rangle \in \mathfrak{B}'} \mathcal{B}$;
(3) $\mathcal{M}_{i-1} \supseteq \mathcal{I}_i$, for each $i \in \{1, \ldots, k\}$, where $\mathcal{M}_0 = \mathcal{U}_{in}$ and $\mathcal{M}_i: G \to \mathbb{N}$ denote the quantities owned after the i-th transformation, i.e., $\mathcal{M}_i = \mathcal{M}_{i-1} \cup \mathcal{O}_i \setminus \mathcal{I}_i$.

Under the *free disposal assumption*, a *solution* to $\mathcal{A}$ is any feasible outcome $\mathfrak{B}'$ such that $\mathcal{M}_k \supseteq \mathcal{U}_{out}$. In the case of *lack of free disposal*, instead, a *solution* to $\mathcal{A}$ is any feasible outcome such that $\mathcal{M}_k = \mathcal{U}_{out}$. □

The *revenue* of the auctioneer with a feasible outcome $\mathfrak{B}'$ is the sum of the payments associated with each atomic bid in it, i.e., the value $\sum_{\langle \mathcal{B}, p, \text{TYPE} \rangle \in \mathfrak{B}'} p$.

An *optimal* solution is a solution providing the auctioneer with the maximum revenue over all the possible solutions.

Example 2.2. Consider again the supply chain presented in Example 1.1. The supply chain can be formalized as an instance $\bar{\mathcal{A}} = \langle \bar{G}, \bar{\mathfrak{B}}, \bar{\mathcal{U}}_{in}, \bar{\mathcal{U}}_{out} \rangle$ where $\bar{G} =$ {frame, brakes, drive train, front wheel, back wheel, seat, handlebars, chain, chainrings, dynamo, bicycle}, $\bar{\mathfrak{B}} = \{\langle\{t_1\}, -2, \text{FULL}\rangle, \langle\{t_2\}, -3, \text{FULL}\rangle, \langle\{t_3\}, -10, \text{FULL}\rangle, \langle\{t_4\}, -17, \text{FULL}\rangle, \langle\{t_5\}, -8, \text{FULL}\rangle, \langle\{t_6\}, -10, \text{FULL}\rangle\}$, $\bar{\mathcal{U}}_{in} =$ {chain, chainrings, frame, brakes}, and $\bar{\mathcal{U}}_{out} =$ {bicycle}. Notice in particular that prices are negative, for they express costs to be payed by the auctioneer.

Under free disposal, the sets $\bar{\mathfrak{B}}' = \{\langle\{t_1\}, -2, \text{FULL}\rangle, \langle\{t_3\}, -10, \text{FULL}\rangle, \langle\{t_5\}, -8, \text{FULL}\rangle, \langle\{t_6\}, -10, \text{FULL}\rangle\}$ and $\bar{\mathfrak{B}}'' = \{\langle\{t_1\}, -2, \text{FULL}\rangle, \langle\{t_4\}, -17, \text{FULL}\rangle, \langle\{t_6\}, -10, \text{FULL}\rangle\}$ are solutions whose associated revenues are -30 and -29, respectively. In fact, $\bar{\mathfrak{B}}''$ is an optimal solution. Instead, under lack of free disposal, $\bar{\mathfrak{B}}'$ is an optimal solution while $\bar{\mathfrak{B}}''$ is not a solution at all. Indeed, by the transformation t_4, the auctioneer obtains a dynamo, for which she might have to pay an additional storage/disposal cost.

Note that each bid in $\bar{\mathfrak{B}}$ consists of exactly one transformation. For a slightly more complex scenario involving combinatorial bids, assume that a novel bidder takes part in the transformation process by offering the whole set $\{t_1, t_4, t_6\}$ for \$11, thereby giving rise to the instance $\langle \bar{G}, \bar{\mathfrak{B}} \cup \{\langle\{t_1, t_4, t_6\}, -11, \text{FULL}\rangle\}, \bar{\mathcal{U}}_{in}, \bar{\mathcal{U}}_{out}\rangle$. Under the free disposal assumption, an optimal solution to the novel instance is the set $\{\langle\{t_1, t_4, t_6\}, -11, \text{FULL}\rangle\}$. Of course, the modification does not alter the optimal solutions under lack of free disposal, given that the novel bid is of type FULL and, thus, accepting it causes implementing t_4.

Note that, if we consider the auction $\langle \bar{G}, \bar{\mathfrak{B}} \cup \{\langle\{t_1, t_4, t_6\}, -11, \text{PARTIAL}\rangle\}, \bar{\mathcal{U}}_{in}, \bar{\mathcal{U}}_{out}\rangle$, then $\{\langle\{t_1, t_4, t_6\}, -11, \text{PARTIAL}\rangle\}$ is an optimal outcome under the free disposal assumption. In absence of free disposal, the optimal outcome is $\{\langle\{t_1, t_4, t_6\}, -11, \text{PARTIAL}\rangle, \langle\{t_3\}, -10, \text{FULL}\rangle, \langle\{t_5\}, -8, \text{FULL}\rangle\}$, which is witnessed by the sequence t_1, t_3, t_5, t_6. $\lhd$

Note that, in some cases, bidders need to communicate more complex bids to the auctioneer, rather than just atomic bids. This calls for adopting an appropriate bidding language. In this paper, we consider the languages based on OR-combinations and XOR-combinations of atomic bids, as defined next (on top of instances over atomic bids). In fact, these languages are natural adaptations in the context of MMUCAs from those languages that are used for combinatorial auctions (see, e.g., [8]). In particular, the OR-language used to be the standard language for combinatorial auctions (see, e.g., [9]), while XOR-combinations of bids have been introduced by [10] in order to obtain a full expressive language, where bidders can report general preferences (both complementarity and substitutability).

Let G be a set of types of goods. A c-*bid* (with $\mathrm{c} \in \{\mathrm{OR}, \mathrm{XOR}\}$) over G is an expression of the form $\mathfrak{L} = \mathbf{b}_1 \ \mathrm{c} \ \mathbf{b}_2 \ \mathrm{c} \ \cdots \ \mathrm{c} \ \mathbf{b}_k$, where $\mathbf{b}_i$ is an atomic bid, for each $1 \leqslant i \leqslant k$. The multi-set $\{\mathbf{b}_1, \ldots, \mathbf{b}_k\}$ of atomic bids is denoted by $bids(\mathfrak{L})$.

Let $\{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$ be a multi-set of OR-bids (resp., XOR-bids), and let $\mathfrak{B}'$ be a multi-set of atomic bids. We say that $\mathfrak{B}'$ *satisfies* $\{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$ if $\mathfrak{B}' \subseteq \bigcup_{\ell=1}^{n} bids(\mathfrak{L}_\ell)$ (resp., $\mathfrak{B}' \subseteq \{\mathbf{b}'_1, \ldots, \mathbf{b}'_n\}$, where $\mathbf{b}'_\ell \in bids(\mathfrak{L}_\ell)$, for each $1 \leqslant \ell \leqslant n$). Thus, with OR-bids (resp., XOR-bids), bidders submit sets of atomic bids and accept to implement any combination of them (resp., at most one of them).

A MMUCA instance *over* c-*bids* is a tuple $\mathcal{A}_\mathrm{C} = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$, where $\{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$ is a multi-set of c-bids. A multi-set $\mathfrak{B}'$ of atomic bids is a *feasible outcome* for $\mathcal{A}_\mathrm{C}$ if $\mathfrak{B}'$ satisfies $\{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$ and is a feasible outcome for the instance $\langle G, \bigcup_{\ell=1}^{n} bids(\mathfrak{L}_\ell), \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ over atomic bids. The notions of solution and optimal solution are then defined analogously to the case of instances over atomic bids.

Example 2.3. In the supply chain discussed in Example 1.1, assume that a bidder communicates that she can implement either the transformation t_1 for \$2 or, alternatively, the transformation t_6 for \$10. This can be formalized via the XOR-bid:

$$\bar{\mathfrak{L}}_1 = \big\langle\{t_1\}, -2, \text{FULL}\big\rangle \ \mathrm{XOR} \ \big\langle\{t_6\}, -10, \text{FULL}\big\rangle.$$

Moreover, consider the XOR-bids $\bar{\mathfrak{L}}_2, \ldots, \bar{\mathfrak{L}}_5$ where $\bar{\mathfrak{L}}_i = \langle\{t_i\}, p_i, \text{FULL}\rangle$ and where p_i is the price of the transformation t_i, for each $2 \leqslant i \leqslant 5$.

These bids give rise to an instance $\bar{\mathcal{A}}_\mathrm{XOR} = \langle \bar{G}, \{\bar{\mathfrak{L}}_1, \ldots, \bar{\mathfrak{L}}_5\}, \bar{\mathcal{U}}_{in}, \bar{\mathcal{U}}_{out}\rangle$ over XOR-bids, where we cannot execute both t_1 and t_6. Given that in order to produce a bicycle, t_6 is mandatory, the constraint implies that the bid over $\{t_1\}$ cannot be accepted. Thus, an optimal solution, under free disposal, has revenue -30 and is witnessed by the sequence t_2, t_4, t_6. Note that nothing changes w.r.t. the case discussed in Example 2.2, if OR-bids were submitted in place of the above XOR-bids. Indeed, given the OR-bid $\langle\{t_1\}, -2, \text{FULL}\rangle$ OR $\langle\{t_6\}, -10, \text{FULL}\rangle$, it is feasible to accept both its atomic bids. $\lhd$

We leave the section by noticing that any MMUCA instance over atomic bids can be viewed as a trivial instance over OR-bids (resp., XOR-bids), where each OR-bid (resp., XOR-bid) actually consists of one atomic bid only.

2.2. Related work

Cerquides et al. [1] introduced Mixed Multi-Unit Combinatorial Auctions (MMUCAs) as an extension of classical combinatorial auctions. They discussed bidding languages, defined the Winner-Determination problem, and pointed out its NP-hardness. Moreover, they provided an algorithmic solution to Winner-Determination, which is based on an integer program (IP) encoding. The solution approach uses binary decision variables for representing the set of transformations that have to be accepted and the order in which they have to be implemented. As a drawback, it exploits a number of variables that is quadratic in the overall number of transformations. Indeed, this formulation has been empirically proved to be useful for solving only small and medium-sized instances [11]. Much of the subsequent research on MMUCAs has been in fact devoted to develop novel solution methods that are able to overcome the limits of this approach.

Two alternative IP formulations have been introduced in [4]. The first one is based on the use of integer decision variables, instead of binary ones, and on some techniques for indexing them. This formulation requires a number of variables that is linear in the number of transformations; however, the constraints needed to refer to the index of a variable by means of another variable are relatively expensive. The second formulation is based on the division of the WINNER-DETERMINATION problem in two subproblems, that (i) determine the multi-set of transformations to accept and (ii) check if this multi-set of transformations is implementable. These two subproblems can be solved by two independent integer programs. The first program looks for multi-sets of transformations that respect the bid constraints, allow to obtain a superset of the desired goods, and maximize the revenue for the auctioneer. The second program looks for a valid sequence of transformations where each transformation in the multi-set returned by the first program is used exactly once. As a matter of fact, the second program might not have a solution for the multi-set of transformations returned as a solution by the first program. Thus, it is sometimes necessary to repeatedly solve the first subproblem, each time eliminating the solution obtained in the previous iteration by adding a constraint.

Improvements in efficiency have been obtained by modifying the original formulation in [1], by adding constraints devoted to code transformation dependencies [3]. In a nutshell, the key observation is that if a transformation requires in input some goods that can be produced only after that another transformation has been executed, then a partial order among these two transformations can be established. Partial orders are subsequently exploited to build a solution template and the search space is pruned by enforcing MMUCA solutions to fulfill such template. By using this novel formulation, an empirical study about the factors (in terms of some topological, problem size, and price-based features) that make MMUCA instances hard to solve has been performed [12].

In a complementary line of research, the formalism of Petri Nets has been adopted in order to obtain more efficient solvers for the WINNER-DETERMINATION problem [13,14]. In particular, an extension of classical Petri Nets, called Weighted Transition Petri Nets (WTPN), has been introduced to encompass the costs associated to transformations, and a new type of reachability problem over them (called MAXSEQ) has been defined. A mapping between MMUCA and WTPN instances has been discussed as well as the correspondence between the WINNER-DETERMINATION problem for MMUCAs and the MAXSEQ problem for the associated WTPN. By means of this approach, the number of decision variables has been reduced from quadratic to linear for all classes of MMUCA instances that can be represented by acyclic WTPNs.

Note that all the methods discussed above have been empirically validated on artificial datasets. Indeed, building generators of syntectic data that are representative of real scenarios is an active area of research [15,11,4,16]. In particular, the MMUCATS tool [16], a test suite for MMUCAs, allows to integrate new WINNER-DETERMINATION algorithms, to evaluate, and to compare them.

Finally, an extension of the MMUCA framework that takes into account some types of time constraints has been recently proposed [19]. This extension supports the specification of partial orderings among the transformations, by relating transformations to absolute time points, and by constraining transformation durations. The WINNER-DETERMINATION problem for this new type of MMUCAs has been defined, and the original IP formulation [1] of the WINNER-DETERMINATION problem has been extended for supporting time constraints. The number of variables in such extended IP formulation remains quadratic in the number of transformations.

3. FEASIBILITY and qualitative restrictions

The FEASIBILITY problem for MMUCAs consists in deciding whether a mixed multi-unit combinatorial auction instance $\mathcal{A}_C = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ over c-bids (with $c \in \{\text{OR}, \text{XOR}\}$) provided as input admits solutions. Note that the problem of deciding the existence of a solution is immaterial for classical combinatorial auctions under the free disposal assumption. Instead, it emerged to be computationally intractable for combinatorial auctions under the lack of free disposal assumption [17]. Moreover, we point out that the problem has been studied for *reverse* auctions, where the auctioneer is trying to buy a certain set of goods, and for *combinatorial exchanges*, which is a generalization of auctions and reverse auctions where participants are allowed to both buy and sell items. In these two latter settings, the problem emerged to be intractable with XOR-bids even under the free disposal assumption [17,18].

In fact, similarly to the case of reverse auctions and combinatorial exchanges, FEASIBILITY makes sense in our context where the auctioneer wants to end up with some desired goods, even under the free disposal assumption. For instance, it is easily seen that there would be no solution in Example 2.3, if the bid $\bar{\mathfrak{L}}_1$ was not submitted to the auctioneer. A crucial problem for MMUCAs is therefore to characterize the computational complexity of FEASIBILITY and, in particular, to single out classes of instances over which computing optimal solutions is feasible in polynomial time. In the following, we shall precisely address this study, by charting the tractability frontier of FEASIBILITY with respect to various "qualitative" properties of the underlying instances. To this end, consider the instance $\mathcal{A}_C = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$, and let $\mathcal{T}_r$ denote the multi-set of all transformations occurring in its bids. Then, let us define the following parameters, where for each good g, the auctioneer making available or requiring g is counted as one virtual transformation producing or consuming it, respectively:

in-var$(\mathcal{A}_C) =$ $\max\{\max_{\langle \mathcal{I}, \mathcal{O}\rangle \in \mathcal{T}_r} |\{g \in G \mid \mathcal{I}(g) > 0\}|, \min\{1, \sum_{g \in G} \mathcal{U}_{out}(g)\}\}$ is the *input variety* of $\mathcal{A}_C$, i.e., the maximum number of types of goods required as input by any given transformation over all possible transformations.

out-var$(\mathcal{A}_C) =$ $\max\{\max_{\langle\mathcal{I},\mathcal{O}\rangle\in\mathcal{T}_r} |\{g \in G \mid \mathcal{O}(g) > 0\}|, \min\{1, \sum_{g\in G}\mathcal{U}_{in}(g)\}\}$ is the *output variety* of $\mathcal{A}_C$, i.e., the maximum number of types of goods produced by any given transformation over all possible transformations.

in-mul$(\mathcal{A}_C) =$ $\max_{g\in G}\{\max_{\langle\mathcal{I},\mathcal{O}\rangle\in\mathcal{T}_r} \mathcal{I}(g), \mathcal{U}_{out}(g)\}$ is the *input multiplicity* of $\mathcal{A}_C$, i.e., the maximum quantity of any good required as input by any given transformation over all possible transformations.

out-mul$(\mathcal{A}_C) =$ $\max_{g\in G}\{\max_{\langle\mathcal{I},\mathcal{O}\rangle\in\mathcal{T}_r} \mathcal{O}(g), \mathcal{U}_{in}(g)\}$ is the *output multiplicity* of $\mathcal{A}_C$, i.e., the maximum quantity of any good produced by any given transformation over all possible transformations.

in-deg$(\mathcal{A}_C) =$ $\max_{g\in G}(|\{\langle\mathcal{I},\mathcal{O}\rangle \in \mathcal{T}_r \mid \mathcal{O}(g) > 0\}| + \min\{1, \mathcal{U}_{in}(g)\})$ is the *input degree* of $\mathcal{A}_C$, i.e., the maximum number of transformations producing any given good over all possible goods.

out-deg$(\mathcal{A}_C) =$ $\max_{g\in G}(|\{\langle\mathcal{I},\mathcal{O}\rangle \in \mathcal{T}_r \mid \mathcal{I}(g) > 0\}| + \min\{1, \mathcal{U}_{out}(g)\})$ is the *output degree* of $\mathcal{A}_C$, i.e., the maximum number of transformations requiring any given good over all possible goods.

Note that all the above parameters depend on the transformations $\mathcal{T}_r$ occurring in the auction, but not on the specific bids defined over them.

Example 3.1. Consider the MMUCA instance $\bar{\mathcal{A}}_{\text{XOR}}$ defined in Example 2.3 and involving the set $\{t_1, \ldots, t_6\}$ of transformations, which are graphically depicted in Fig. 1. By inspecting the figure, we can see that *in-var*$(\bar{\mathcal{A}}_{\text{XOR}}) = 7$ (because of t_6), *out-var*$(\bar{\mathcal{A}}_{\text{XOR}}) = 5$ (because of t_4), *in-mul*$(\bar{\mathcal{A}}_{\text{XOR}}) =$ *out-mul*$(\bar{\mathcal{A}}_{\text{XOR}}) = 1$ (since all transformations require and produce just one item of each type of good, and since one item at most of each type of good is initially available or finally requested by the auctioneer), *in-deg*$(\bar{\mathcal{A}}_{\text{XOR}}) = 2$ (because, e.g., of seat that is produced by t_3 and t_4), and *out-deg*$(\bar{\mathcal{A}}_{\text{XOR}}) = 2$ (because, e.g., of chain that is required by t_1 and t_2). $\lhd$

In the following, the notation $\mathcal{C}_C(\mathtt{iv}, \mathtt{ov}, \mathtt{im}, \mathtt{om}, \mathtt{id}, \mathtt{od})$ will be used to indicate the class of all instances $\mathcal{A}_C$ such that: *in-var*$(\mathcal{A}_C) \leqslant \mathtt{iv}$, *out-var*$(\mathcal{A}_C) \leqslant \mathtt{ov}$, *in-mul*$(\mathcal{A}_C) \leqslant \mathtt{im}$, *out-mul*$(\mathcal{A}_C) \leqslant \mathtt{om}$, *in-deg*$(\mathcal{A}_C) \leqslant \mathtt{id}$, and *out-deg*$(\mathcal{A}_C) \leqslant \mathtt{od}$. When focusing on instances with atomic bids only, the subscript 'C' will be omitted. Moreover, we shall use the symbol ∞ to denote that no bound is issued on some given parameter.

Summary of results. A detailed analysis of the computational complexity arising with the FEASIBILITY problem over several classes $\mathcal{C}_C(\mathtt{iv}, \mathtt{ov}, \mathtt{im}, \mathtt{om}, \mathtt{id}, \mathtt{od})$ of instances is reported in the rest of this section. In particular, the complexity of FEASIBILITY is analyzed by considering two different scenarios:

1. First, we analyze the case where each atomic bid $\langle\mathcal{B}, p, \text{TYPE}\rangle$ involved in the definition of the MMUCA instance is such that $|\mathcal{B}| = 1$, i.e., such that all atomic bids are defined over singleton sets of transformations. This is, for instance, the case of Example 2.2. In fact, this scenario is peculiar for MMUCAs and its analysis is therefore crucial to single out the source of intractability that can be hidden in the interactions occurring in production processes.
2. Then, we consider the general case where no restriction is imposed over the bids involved in the instance. Note that this scenario is essentially equivalent to the one in (1), whenever all bids are of TYPE = PARTIAL. Indeed, as far as the FEASIBILITY problem is concerned, any atomic bid $\langle\mathcal{B}, p, \text{TYPE}\rangle$ with TYPE = PARTIAL and where $|\mathcal{B}| = k$ can be transparently substituted with k fresh atomic bids (whose associated payments are irrelevant), each one being defined over a different transformation taken from $\mathcal{B}$. Therefore, we shall only focus to the case where there is at least one bid of TYPE = FULL.

A summary of the complexity results for the case where bids are defined over singleton sets of transformations is reported in Fig. 2. Note that the expressiveness of XOR-bids [1] is payed in terms of smaller tractability islands. Moreover, note that the complexity of OR-bids and atomic bids coincides, with hardness results being actually given over auctions that use atomic bids only. Finally, note that the lack of free disposal represents an additional source of intractability.

A summary of the complexity results arising with bids defined over arbitrary sets of transformations (and with at least one bid with TYPE = FULL) is reported in Fig. 3. Of course, all hardness results provided for singleton sets of transformations still hold in this more general setting. In fact, most of the results in Fig. 2 have been strengthened, thus showing that introducing the possibility of defining bids over arbitrary sets of transformations makes problems substantially harder than before.

The rest of the section is devoted to provide detailed proofs of such complexity results. Actually, membership in NP has been show in [1] for arbitrary MMUCA instances. Thus, we shall either just provide NP-hardness results, or single out islands of tractability.

3.1. Hard instances for atomic bids over singleton sets of transformations

Let us start by considering MMUCA instances over atomic bids, and by providing hardness results that still hold when bids are defined over singleton sets of transformations. In this scenario, since we are interested in the FEASIBILITY problem, we can get rid, w.l.o.g., of the type of the bids and of their associated payments. Thus, any bid can be equivalently viewed as a transformation, and a MMUCA instance $\mathcal{A}$ can be defined as a tuple $\langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$, where $\mathcal{T}$ is a multi-set of transformations. A solution is then a sequence of transformations taken from $\mathcal{T}$ and satisfying condition (3) in Definition 2.1.

OR-*bids* ≡ *atomic bids*						
iv	ov	im	om	id	od	Free disposal
1	1	1	1	∞	∞	in P [Thm. 3.13]
∞	∞	∞	∞	∞	1	in P [Thm. 3.14]
∞	∞	∞	∞	1	∞	in P [Thm. 3.15]
2	1	1	1	2	2	NP-complete [Thm. 3.5]
1	2	1	1	2	2	NP-complete [Thm. 3.3]
1	1	2	1	2	2	NP-complete [Thm. 3.6]
1	1	1	2	2	2	NP-complete [Thm. 3.4]

XOR-*bids*						
iv	ov	im	om	id	od	Free disposal
∞	∞	∞	∞	1	∞	in P [Thm. 3.16]
1	1	1	1	2	1	NP-complete [Thm. 3.11]

OR-*bids* ≡ *atomic bids*						
iv	ov	im	om	id	od	Lack of free disposal
1	1	1	1	∞	∞	in P [Thm. 3.13]
∞	∞	∞	∞	∞	1	in P [Thm. 3.14]
1	1	∞	∞	1	∞	NP-complete [Thm. 3.10]
2	1	1	1	1	2	NP-complete [Thm.3.9]
1	2	1	1	2	2	NP-complete [Thm.3.8]
1	1	2	1	2	2	NP-complete [Thm.3.8]
1	1	1	2	2	2	NP-complete [Thm.3.8]

XOR-*bids*						
iv	ov	im	om	id	od	Lack of free disposal
∞	∞	∞	∞	1	1	in P [Thm. 3.17]
1	1	∞	∞	1	∞	NP-complete [Thm. 3.12]
1	1	1	1	2	1	NP-complete [Thm. 3.12]
2	1	1	1	1	2	NP-complete [Thm. 3.12]

Fig. 2. FEASIBILITY and qualitative restrictions. Summary of results for bids over singleton sets of transformations.

We start by analyzing the case when the free disposal assumption holds. Hardness results for this setting are next provided via reductions from the problem of checking the SATISFIABILITY of Boolean formulas in conjunctive normal form. Recall that deciding whether a Boolean formula in conjunctive normal form $\Phi = c_1 \wedge \cdots \wedge c_m$ over the variables $X_1, \ldots, X_n$ is satisfiable, i.e., deciding whether there exists a truth assignment to the variables making each clause c_j true, is NP-hard [20]. In fact, to simplify the following reductions, we find it useful to state the intractability of a specific class of Boolean formulas, whose proof is reported in Appendix A.

Lemma 3.2. SATISFIABILITY *is* NP-*hard, even on classes of Boolean formulas in conjunctive normal form where each variable occurs positively in two clauses and negatively in another, and where each clause contains three literals at most.*

Theorem 3.3. FEASIBILITY *is* NP-*hard under the free disposal assumption, even when restricted to the class* $\mathcal{C}(1, 2, 1, 1, 2, 2)$.

Proof. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula over the variables $X_1, \ldots, X_n$ satisfying the conditions in Lemma 3.2, and let $\mathcal{A}(\Phi) = \langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$ be the MMUCA instance over atomic bids (defined over singleton sets of transformations) such that: $G = \bigcup_{i=1}^{n} \{X_i, X_i^T, X_i^F\} \cup \{c_1, \ldots, c_m\} \cup \{c_j^i \mid X_i \text{ occurs in } c_j\}$, $\mathcal{U}_{in} = \{X_1, \ldots, X_n\}$, $\mathcal{U}_{out} = \{c_1, \ldots, c_m\}$, and $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$. In particular, for each variable X_i occurring positively in the clauses c_α and c_β while occurring negatively in c_γ, the set $\mathcal{T}_i$ consists of the set of transformations $\{\langle \{X_i\}, \{X_i^T\} \rangle, \langle \{X_i\}, \{X_i^F\} \rangle, \langle \{X_i^T\}, \{c_\alpha^i, c_\beta^i\} \rangle, \langle \{X_i^F\}, \{c_\gamma^i\} \rangle, \langle \{c_\alpha^i\}, \{c_\alpha\} \rangle, \langle \{c_\beta^i\}, \{c_\beta\} \rangle$, and $\langle \{c_\gamma^i\}, \{c_\gamma\} \rangle\}$. An illustration of $\mathcal{T}_i$ is reported in Fig. 4.

Observe that two transformations occur in $\mathcal{T}_i$ requiring X_i as input, one that produces X_i^T and another that produces X_i^F. These transformations are meant to encode the selection of a truth value assignment to the variable X_i and, in fact, they are mutually exclusive in any solution, since there is just one copy of X_i in $\mathcal{U}_{in}$.

OR-*bids* ≡ *atomic bids*						
iv	ov	im	om	id	od	Free disposal
∞	∞	∞	∞	∞	1	in P [Thm. 3.20]
∞	∞	∞	∞	1	∞	in P [Thm. 3.19]
1	1	1	1	2	2	NP-complete [Thm. 3.18]

XOR-*bids*						
iv	ov	im	om	id	od	Free disposal
∞	∞	∞	∞	1	∞	in P [Thm. 3.19]
1	1	1	1	2	1	NP-complete [Thm. 3.11]

OR-*bids* ≡ *atomic bids*						
iv	ov	im	om	id	od	Lack of free disposal
∞	∞	∞	∞	∞	1	in P [Thm. 3.20]
1	1	1	1	1	2	NP-complete [Thm. 3.18]

XOR-*bids*						
iv	ov	im	om	id	od	Lack of free disposal
∞	∞	∞	∞	1	1	in P [Thm. 3.21]
1	1	1	1	2	1	NP-complete [Thm. 3.12]
1	1	1	1	1	2	NP-complete [Thm. 3.18]

Fig. 3. Feasibility and qualitative restrictions. Summary of results for bids over arbitrary (i.e., not necessarily singleton) sets of transformations.

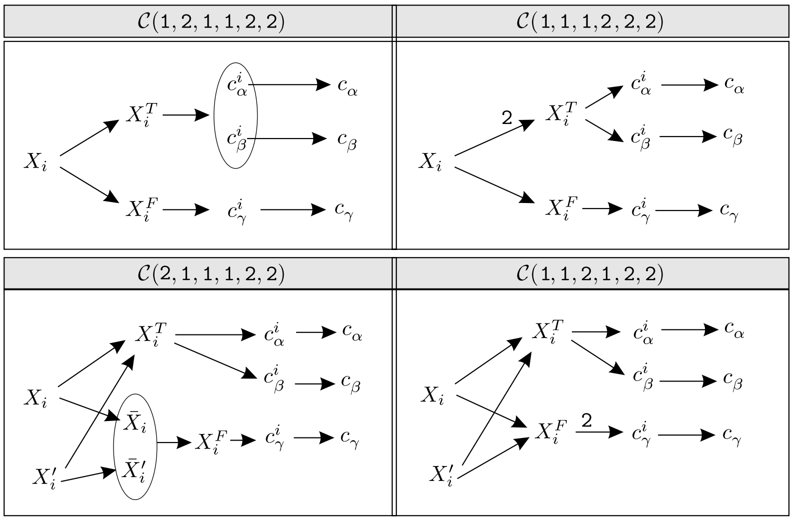

Fig. 4. Hardness results over atomic bids: Reductions in the proofs of Theorems 3.3, 3.4, 3.5, and 3.6.

Based on the above observation, we now claim: *Φ is satisfiable $\Leftrightarrow$ $\mathcal{A}(\Phi)$ has a solution.*

($\Rightarrow$) Let $\mathbf{x}$ be a satisfying assignment of Φ, and consider the sequences of transformations σ_i, for each $1 \leqslant i \leqslant n$, such that $\sigma_i = \langle\{X_i\}, \{X_i^T\}\rangle, \langle\{X_i^T\}, \{c_\alpha^i, c_\beta^i\}\rangle, \langle\{c_\alpha^i\}, \{c_\alpha\}\rangle, \langle\{c_\beta^i\}, \{c_\beta\}\rangle$ if X_i evaluates true in $\mathbf{x}$, and such that $\sigma_i = \langle\{X_i\}, \{X_i^F\}\rangle, \langle\{X_i^F\}, \{c_\gamma^i\}\rangle, \langle\{c_\gamma^i\}, \{c_\gamma\}\rangle$ if X_i evaluates false. By construction, the sequence $\sigma_1, \ldots, \sigma_n$ obtained as their concatenation satisfies condition (3) in Definition 2.1 and leads to the desired final configuration. Thus, $\sigma_1, \ldots, \sigma_n$ is a solution to $\mathcal{A}(\Phi)$.

($\Leftarrow$) Let σ be a solution to $\mathcal{A}(\Phi)$. Then, for each clause c_j, with $1 \leqslant j \leqslant m$, σ executed a set of transformations producing as an intermediate good either X_i^T or X_i^F, for some variable X_i occurring in c_j: Let $\mathbf{x}$ be the (possibly partial) truth assignment where each such variable X_i evaluates true (resp., false) if X_i^T (resp., X_i^F) is the produced good. It is immediate to check that $\mathbf{x}$ is a satisfying assignment of Φ.

To conclude the proof, observe that $\mathcal{A}(\Phi)$ belongs to $\mathcal{C}(1,2,1,1,3,2)$, but not necessarily to $\mathcal{C}(1,2,1,1,2,2)$ since a clause c_j may contain three variables $X_i, X_{i'}$, and $X_{i''}$. In order to face this case, we can add one further good $\bar{c}_j$, and replace the two transformations $\langle\{c_j^{i'}\},\{c_j\}\rangle$ and $\langle\{c_j^{i''}\},\{c_j\}\rangle$ with the transformations: $\langle\{c_j^{i'}\},\{\bar{c}_j\}\rangle$, $\langle\{c_j^{i''}\},\{\bar{c}_j\}\rangle$, and $\langle\{\bar{c}_j\},\{c_j\}\rangle$. Clearly, the resulting auction has input degree equals to 2, while keeping unchanged the values of all the other parameters and the properties of the reduction. □

The following three results are based on simple adaptations of the reduction in Theorem 3.3.

Theorem 3.4. Feasibility *is* NP-*hard under the free disposal assumption, even when restricted to the class* $\mathcal{C}(1,1,1,2,2,2)$.

Proof. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula over the variables $X_1,\ldots,X_n$ satisfying the conditions in Lemma 3.2. Let $\mathcal{A}^1(\Phi)$ be the auction that is obtained from $\mathcal{A}(\Phi)$ (defined in the proof of Theorem 3.3) by substituting the two transformations $\langle\{X_i\},\{X_i^T\}\rangle$ and $\langle\{X_i^T\},\{c_\alpha^i, c_\beta^i\}\rangle$, with the transformations $\langle\{X_i\},\{X_i^T, X_i^T\}\rangle$, $\langle\{X_i^T\},\{c_\alpha^i\}\rangle$, $\langle\{X_i^T\},\{c_\beta^i\}\rangle$, for each $1 \leqslant i \leqslant n$. Thus, X_i can generate two copies of X_i^T, each one enabling one of the two clauses where X_i positively occurs. See Fig. 4 for an illustration. Eventually, each of the two transformations $\langle\{X_i\},\{X_i^T, X_i^T\}\rangle$ and the transformation $\langle\{X_i\},\{X_i^F\}\rangle$ are still mutually exclusive in any sequence of transformations satisfying condition (3) in Definition 2.1. Thus, the line of reasoning in the proof of Theorem 3.3 still applies (in particular, note that the case where *in-deg*$(\mathcal{A}^1(\Phi)) = 3$ can be reduced to a case where *in-deg*$(\mathcal{A}^1(\Phi)) = 2$), and we can conclude that Φ is satisfiable $\Leftrightarrow$ $\mathcal{A}^1(\Phi)$ has a solution. □

Theorem 3.5. Feasibility *is* NP-*hard under the free disposal assumption, even when restricted to the class* $\mathcal{C}(2,1,1,1,2,2)$.

Proof. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula over the variables $X_1,\ldots,X_n$ satisfying the conditions in Lemma 3.2. Consider the auction $\mathcal{A}^2(\Phi) = \langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ such that: $G = \bigcup_{i=1}^n \{X_i, X_i', X_i^T, \overline{X}_i, \overline{X}_i', X_i^F\} \cup \{c_1,\ldots,c_m\} \cup \{c_j^i \mid X_i \text{ occurs in } c_j\}$, $\mathcal{U}_{in} = \{X_1,\ldots,X_n,X_1',\ldots,X_n'\}$, $\mathcal{U}_{out} = \{c_1,\ldots,c_m\}$, and $\mathcal{T} = \bigcup_{i=1}^n \mathcal{T}_i$. In particular, for each variable X_i occurring positively in the clauses c_α and c_β while occurring negatively in c_γ, $\mathcal{T}_i$ is the set $\{\langle\{X_i\},\{X_i^T\}\rangle, \langle\{X_i\},\{\overline{X}_i\}\rangle, \langle\{X_i'\},\{X_i^T\}\rangle, \langle\{X_i'\},\{\overline{X}_i'\}\rangle, \langle\{\overline{X}_i, \overline{X}_i'\},\{X_i^F\}\rangle, \langle\{X_i^T\},\{c_\alpha^i\}\rangle, \langle\{X_i^T\},\{c_\beta^i\}\rangle, \langle\{X_i^F\},\{c_\gamma^i\}\rangle, \langle\{c_\alpha^i\},\{c_\alpha\}\rangle, \langle\{c_\beta^i\},\{c_\beta\}\rangle, \langle\{c_\gamma^i\},\{c_\gamma\}\rangle\}$. An illustration of $\mathcal{T}_i$ is reported in Fig. 4.

Observe that for each variable X_i, the transformation $\langle\{X_i\},\{X_i^T\}\rangle$ is mutually exclusive with $\langle\{X_i\},\{\overline{X}_i\}\rangle$, and that $\langle\{X_i'\},\{X_i^T\}\rangle$ is mutually exclusive with $\langle\{X_i'\},\{\overline{X}_i'\}\rangle$. Thus, the application of a transformation in $\{\langle\{X_i\},\{X_i^T\}\rangle, \langle\{X_i'\},\{X_i^T\}\rangle\}$ is mutually exclusive with the application of $\langle\{\overline{X}_i, \overline{X}_i'\},\{X_i^F\}\rangle$, thereby encoding a selection of a truth value assignment to the variable X_i. Thus, the line of reasoning in the proof of Theorem 3.4 still applies (in particular, note that the case where *in-deg*$(\mathcal{A}^2(\Phi)) = 3$ can be reduced to a case where *in-deg*$(\mathcal{A}^2(\Phi)) = 2$), so that it possible to produce a superset of $\{c_1,\ldots,c_m\}$ if, and only if, all the various selected transformations encode a satisfying assignment of Φ. That is, Φ is satisfiable $\Leftrightarrow$ $\mathcal{A}^2(\Phi)$ has a solution. □

Theorem 3.6. Feasibility *is* NP-*hard under the free disposal assumption, even when restricted to the class* $\mathcal{C}(1,1,2,1,2,2)$.

Proof. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula over the variables $X_1,\ldots,X_n$ satisfying the conditions in Lemma 3.2. Let $\mathcal{A}^3(\Phi)$ be the auction obtained from $\mathcal{A}^2(\Phi)$ (defined in the proof of Theorem 3.5) by substituting the transformations $\langle\{X_i\},\{\overline{X}_i\}\rangle$, $\langle\{X_i'\},\{\overline{X}_i'\}\rangle$, $\langle\{\overline{X}_i, \overline{X}_i'\},\{X_i^F\}\rangle$ and $\langle\{X_i^F\},\{c_\gamma^i\}\rangle$, with the transformations $\langle\{X_i\},\{X_i^F\}\rangle$, $\langle\{X_i'\},\{X_i^F\}\rangle$ and $\langle\{X_i^F, X_i^F\},\{c_\gamma^i\}\rangle$, for each $1 \leqslant i \leqslant n$. Basically, X_i and X_i' may now generate two copies of X_i^F, together enabling the clause where X_i negatively occurs. See Fig. 4 for an illustration. In fact, the transformation $\langle\langle\{X_i^F, X_i^F\},\{c_\gamma^i\}\rangle\rangle$ can be applied if, and only if, none of the transformations in $\{\langle\{X_i^T\},\{c_\alpha^i\}\rangle, \langle\{X_i^T\},\{c_\beta^i\}\rangle\}$ is applied. Thus, the line of reasoning in the proof of Theorem 3.3 can still be used (again, the case where *in-deg*$(\mathcal{A}^3(\Phi)) = 3$ can be reduced to a case where *in-deg*$(\mathcal{A}^3(\Phi)) = 2$), and we have that Φ is satisfiable $\Leftrightarrow$ $\mathcal{A}^3(\Phi)$ has a solution. □

In order to complete the picture, we need now to consider the problem under the lack of free disposal assumption.

In fact, in the proofs of Theorems 3.3, 3.4, 3.5, and 3.6, the careful reader may have noticed that if the formula Φ is satisfiable, then there might be solutions to the MMUCA instance (built based on Φ) producing more than one copy of some output good. Indeed, this happens if the same clause is satisfied by two different variables in a satisfying truth assignment. To avoid these cases and smoothly apply the above proofs under the lack of free disposal assumption, we shall use a reduction from the problem of deciding whether, given a Boolean formula Φ, there is a satisfying assignment such that, for each clause, exactly one literal evaluates true. The problem, which we call exact-1 Satisfiability, remains NP-hard even if each clause exactly contains three literals (problem ONE-IN-THREE 3SAT in [21]). Below we state the hardness of a variant of this problem, whose complexity is proven in Appendix A and which applies to the Boolean formulas defined in Lemma 3.2.

Lemma 3.7. EXACT-1 SATISFIABILITY *is* NP*-hard, even on classes of Boolean formulas in conjunctive normal form where each variable occurs positively in two clauses and negatively in another, and where each clause contains three literals at most.*

In the light of the above lemma, by inspecting the proofs of Theorems 3.3, 3.4, 3.5, and 3.6, we obtain that, given a formula Φ, there is a satisfying assignment such that, for each clause, exactly one literal evaluates true (i.e., EXACT-1 SATISFIABILITY has a solution over Φ) if, and only if, the corresponding MMUCA has a solution under the lack of free disposal. Thus, the following can be straightforwardly derived.

Theorem 3.8. FEASIBILITY *is* NP*-hard under the lack of free disposal assumption, even when restricted to the classes* $\mathcal{C}(2,1,1,1,2,2)$, $\mathcal{C}(1,2,1,1,2,2)$, $\mathcal{C}(1,1,2,1,2,2)$*, and* $\mathcal{C}(1,1,1,2,2,2)$.

In fact, Theorem 3.8 can be further strengthened in the two directions which are next analyzed.

Theorem 3.9. FEASIBILITY *is* NP*-hard under the lack of free disposal assumption, even when restricted to the class* $\mathcal{C}(2,1,1,1,1,2)$.

Proof. The reduction is again from the EXACT-1 SATISFIABILITY problem. We recall first that EXACT-1 SATISFIABILITY is known to be NP-hard even on classes of Boolean formulas in conjunctive normal form where not only each clause contains exactly three literals (problem ONE-IN-THREE 3SAT in [21]), but also if all variables occur positively in the formula [21]. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula over the variables $X_1,\ldots,X_n$ satisfying the conditions stated above. Consider the MMUCA instance $\mathcal{A}^4(\Phi) = \langle G, \mathcal{T}, \mathcal{U}_{in}, \emptyset\rangle$ such that: $G = \{X_1,\ldots,X_n\} \cup \bigcup_{i=1}^{n}\{c_j^i \mid X_i \text{ occurs in } c_j\} \cup \{c_1,\ldots,c_m\}$, $\mathcal{U}_{in} = \{c_1,\ldots,c_m\}$, and $\mathcal{T} = \bigcup_{i=1}^{n}\mathcal{T}_i \cup \bigcup_{j=1}^{m}\mathcal{T}'_j$. In particular, for each variable X_i, $\mathcal{T}_i$ is the set $\{\langle\{c_j^i \mid X_i \text{ occurs in } c_j\}, \{X_i\}\rangle, \langle\{X_i\}, \emptyset\rangle\}$. And, for each clause c_j, $\mathcal{T}'_j$ is the set $\{\langle\{c_j\}, \{c_j^i\}\rangle \mid X_i \text{ occurs in } c_j\}\}$.

Note that the transformations in $\mathcal{T}'_j$ are meant to encode the selection of a variable satisfying the clause and they are mutually exclusive with each other, since there is just one copy of c_j in $\mathcal{A}^4(\Phi)$. On the other hand, the transformations in $\mathcal{T}_i$ imply that X_i can be derived if, and only if, c_j^i is derived for each clause c_j where X_i occurs. In fact, whenever X_i is produced, it can be consumed by the transformation $\langle\{X_i\}, \emptyset\rangle$.

Based on these observations, we now claim that EXACT-1 SATISFIABILITY *has a solution over* $\Phi \Leftrightarrow \mathcal{A}^4(\Phi)$ *has a solution under the lack of free disposal assumption.*

(⇒) Let $\mathbf{x}$ be a satisfying assignment of Φ such that, for each clause, exactly one variable evaluates true. In particular, for a clause c_j, let $\mathbf{x}(j)$ denote the index of the variable in c_j that evaluates true in $\mathbf{x}$. Consider now the set of transformations $S_j = \{\langle\{c_j\}, \{c_j^{\mathbf{x}(j)}\}\rangle, \langle\{c_{j'}^{\mathbf{x}(j)} \mid X_{\mathbf{x}(j)} \text{ occurs in } c_{j'}\}, \{X_{\mathbf{x}(j)}\}\rangle, \langle\{X_{\mathbf{x}(j)}\}, \emptyset\rangle\}$. Note that if c_j and $c_{j'}$ are two clauses containing a variable X_i evaluating true in $\mathbf{x}$, then it must be the case that $i = \mathbf{x}(j) = \mathbf{x}(j')$. It follows that we can order the transformations in the set $\bigcup_{j=1}^{m} S_j$ in a way satisfying condition (3) in Definition 2.1. Note that after the application of such a sequence of transformations, no good remains to the auctioneer.

(⇐) Let σ be a solution to $\mathcal{A}^4(\Phi)$. In order to consume all goods initially owned due to the lack of free disposal, σ must include (as a postfix) a number of transformations of the form $\langle\{X_i\}, \emptyset\rangle$. Moreover, for each transformation consuming X_i, we have in turn that σ includes the transformation $\langle\{c_j^i \mid X_i \text{ occurs } c_j\}, \{X_i\}\rangle$. It follows that the truth assignment $\mathbf{x}$ where each variable X_i evaluates true if, and only if, X_i is an intermediate good produced by σ is a satisfying assignment of Φ. Moreover, as transformations in $\mathcal{T}'_j$ are mutually exclusive, $\mathbf{x}$ is such that, for each clause, exactly one literal evaluates true.

To conclude the proof, observe that $\mathcal{A}^4(\Phi)$ belongs to $\mathcal{C}(3,1,1,1,1,k)$, since a clause c_j contain three variables (and, thus, *out-deg*$(\mathcal{A}^4(\Phi)) = 3$), and since the same variable can occur in k clauses (and, thus, *in-var*$(\mathcal{A}^4(\Phi)) = k$).

For the former case, let c_j be a clause containing the variables X_i, $X_{i'}$, and $X_{i''}$. Then, we can add one fresh good $\bar{c}_j$, and replace the transformations in $\mathcal{T}'_j$ with $\langle\{c_j\}, \{c_j^i\}\rangle$, $\langle\{c_j\}, \{\bar{c}_j\}\rangle$, $\langle\{\bar{c}_j\}, \{c_j^{i'}\}\rangle$, $\langle\{\bar{c}_j\}, \{c_j^{i''}\}\rangle$. The output degree in the transformed auction is 2. Instead, for the latter case, if a variable X_i occurs in the clauses $c_{j1},\ldots,c_{jk}$, then we can add the fresh goods $X_i^1,\ldots,X_i^k$ and substitute the transformations in $\mathcal{T}_i$ with $\langle\{c_{j_1}^i\}, \{X_i^1\}\rangle$, $\langle\{X_i^1, c_{j_2}^i\}, \{X_i^2\}\rangle, \ldots, \langle\{X_i^{k-1}, c_{j_k}^i\}, \{X_i^k\}\rangle$, $\langle\{X_i^k\}, \{X_i\}\rangle$, $\langle\{X_i\}, \emptyset\rangle$. The input degree is 2. All the other properties of the reduction remain unchanged. □

Theorem 3.10. FEASIBILITY *is* NP*-hard under the lack of free disposal assumption, even when restricted to the class* $\mathcal{C}(1,1,\infty,\infty,1,\infty)$.

Proof. Deciding whether there is a way to partition a bag $S = \{s_1, s_2, \ldots, s_n\}$ of integers into two bags S_1 and S_2 such that the sum of the numbers in S_1 equals the sum of the numbers in S_2 is the NP-complete PARTITION problem [21].

Let $m = \sum_{i=1}^{n} s_i$, and consider the MMUCA instance $\mathcal{A}(S) = \langle G, \mathcal{T}, \mathcal{U}_{in}, \emptyset\rangle$ such that: $G = \{\alpha\}$, $\mathcal{U}_{in}(\alpha) = m/2$, and $\mathcal{T} = \bigcup_{i=1}^{n}\{\langle\bigcup_{i=1}^{s_i}\{\alpha\}, \{\}\rangle\}$. Note that there is no bound on the output multiplicity of $\mathcal{A}(S)$, since $\mathcal{U}_{in}(\alpha) = m/2$.

It is immediate to check that, under the lack of free disposal, there is a solution to $\mathcal{A}(S)$ if, and only if, Partition has a solution on input S. Indeed, solutions to $\mathcal{A}(S)$ one-to-one correspond with sets of transformations consuming exactly all the input goods, and thus such that the sum of their associated input requirements equals to $m/2$. □

For the sake of completeness, note that the reduction in the above proof is from the Partition problem, which is feasible in polynomial time if all integers are assumed to be given in unary (formally, the problem is weakly NP-hard). Assessing whether a strong NP-hardness result can be achieved is an interesting technical question.

3.2. *Hard instances for* XOR-*bids over singleton sets of transformations*

Let us now move to analyze the case of XOR-bids, by providing hardness results that still hold under the simple scenario where all the bids are defined over singleton sets of transformations. W.l.o.g. (given our interest in the Feasibility problem and given the focus on singleton sets of transformations), in the following, we shall simply view any XOR-bid $\mathfrak{L}$ as an expression t_1 XOR t_2 XOR ... XOR t_k, where t_i is a transformation (rather than an atomic bid), for each $1 \leqslant i \leqslant k$.

We start with the case of free disposal.

Theorem 3.11. Feasibility *is* NP-*hard under the free disposal assumption, even when restricted to the class* $\mathcal{C}_{\text{XOR}}(1, 1, 1, 1, 2, 1)$.

Proof. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula over the variables $X_1, \ldots, X_n$ satisfying the conditions in Lemma 3.2. Consider the MMUCA instance over XOR-bids $\mathcal{A}^5(\Phi) = \langle G, \{\mathfrak{L}_1^a, \ldots, \mathfrak{L}_n^a \mid a \in \{1, \ldots, 4\}\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ such that: $G = \{c_1, \ldots, c_m\} \cup \{c_j^i \mid X_i \text{ occurs in } c_j\}$, $\mathcal{U}_{in} = \{\}$, $\mathcal{U}_{out} = \{c_1, \ldots, c_m\}$, and where $\{\mathfrak{L}_1^a, \ldots, \mathfrak{L}_n^a \mid a \in \{1, \ldots, 4\}\}$ is a set of XOR-bids defined as follows. For each variable X_i that occurs positively in c_α and c_β while occurring negatively in c_γ, let us define $\mathfrak{L}_i^1 = \langle\{\}, \{c_\alpha^i\}\rangle$ XOR $\langle\{\}, \{c_\gamma^i\}\rangle$, $\mathfrak{L}_i^2 = \langle\{c_\gamma^i\}, \{c_\gamma\}\rangle$ XOR $\langle\{\}, \{c_\beta^i\}\rangle$, $\mathfrak{L}_i^3 = \langle\{c_\alpha^i\}, \{c_\alpha\}\rangle$, and $\mathfrak{L}_i^4 = \langle\{c_\beta^i\}, \{c_\beta\}\rangle$.

We now claim that *Φ is satisfiable $\Leftrightarrow$ $\mathcal{A}(\Phi)$ has a solution.*

($\Rightarrow$) Let $\mathbf{x}$ be a satisfying assignment of Φ, and consider the sequences of transformations σ_i, for each $1 \leqslant i \leqslant n$, such that $\sigma_i = \langle\{\}, \{c_\alpha^i\}\rangle, \langle\{\}, \{c_\beta^i\}\rangle, \langle\{c_\alpha^i\}, \{c_\alpha\}\rangle, \langle\{c_\beta^i\}, \{c_\beta\}\rangle$ if X_i evaluates true in $\mathbf{x}$, and such that $\sigma_i = \langle\{\}, \{c_\gamma^i\}\rangle$, $\langle\{c_\gamma^i\}, \{c_\gamma\}\rangle$ if X_i evaluates false. By construction, the sequence $\sigma_1, \ldots, \sigma_n$ obtained as their concatenation satisfies condition (3) in Definition 2.1 and leads to the desired final configuration. Moreover, note that the set of all the bids in such sequence satisfies $\{\mathfrak{L}_i^1, \mathfrak{L}_i^2, \mathfrak{L}_i^3, \mathfrak{L}_i^4\}$, for each $1 \leqslant i \leqslant n$. Thus, $\sigma_1, \ldots, \sigma_n$ is a solution to $\mathcal{A}^5(\Phi)$.

($\Leftarrow$) Let σ be a solution to $\mathcal{A}^5(\Phi)$. Note that, for each variable X_i that occurs positively in c_α and c_β while occurring negatively in c_γ, σ cannot execute the transformation $\langle\{c_\gamma^i\}, \{c_\gamma\}\rangle$ together with any of the transformations in $\{\langle\{c_\alpha^i\}, \{c_\alpha\}\rangle, \langle\{c_\beta^i\}, \{c_\beta\}\rangle\}$. Thus, let $\mathbf{x}$ be the truth assignment where each variable X_i evaluates false if, and only if, $\langle\{c_\gamma^i\}, \{c_\gamma\}\rangle$ is executed in σ. By construction, it is immediate to check that $\mathbf{x}$ is a satisfying assignment of Φ.

To conclude the proof, note that we may have *in-deg*$(\mathcal{A}^5(\Phi)) = 3$ because of some clause containing three variables. Normalization can be carried out along the line discussed in the proof of Theorem 3.3, in order to guarantee that the input degree of the transformed instance is 2. □

Then, the picture is easily completed for the case of lack of free disposal.

Theorem 3.12. Feasibility *is* NP-*hard under the lack of free disposal, even when restricted to the classes* $\mathcal{C}_{\text{XOR}}(1, 1, 1, 1, 2, 1)$, $\mathcal{C}_{\text{XOR}}(2, 1, 1, 1, 1, 2)$, *and* $\mathcal{C}_{\text{XOR}}(1, 1, \infty, \infty, 1, \infty)$.

Proof. The fact that Feasibility is NP-hard on the class $\mathcal{C}_{\text{XOR}}(1, 1, 1, 1, 2, 1)$ under the lack of free disposal follows by inspection of the proof of Theorem 3.11 and by exploiting a reduction from the problem in Lemma 3.7—see the arguments used in the previous section for Theorem 3.8. Hardness over the classes $\mathcal{C}_{\text{XOR}}(2, 1, 1, 1, 1, 2)$ and $\mathcal{C}_{\text{XOR}}(1, 1, \infty, 1, 1, \infty)$ follows from Theorem 3.9 and Theorem 3.10, since $\mathcal{C}_{\text{XOR}}(2, 1, 1, 1, 1, 2) \supset \mathcal{C}(2, 1, 1, 1, 1, 2)$ and $\mathcal{C}_{\text{XOR}}(1, 1, \infty, \infty, 1, \infty) \supset \mathcal{C}(1, 1, \infty, \infty, 1, \infty)$, respectively. □

3.3. *Tractable instances for* OR-*bids over singleton sets of transformations*

Let us now focus on proving tractability results, by starting with the case of OR-bids defined over singleton sets of transformations. Before presenting this analysis, we note that if $\mathcal{A}_{\text{OR}} = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ is a MMUCA instance where $\{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$ is a multi-set of OR-bids, then deciding the existence of a solution to $\mathcal{A}_{\text{OR}}$ just reduces to deciding the existence of a solution to the instance $\langle G, \bigcup_{\ell=1}^{n} bids(\mathfrak{L}_\ell), \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ over atomic bids. Indeed, the satisfiability of OR-bids is

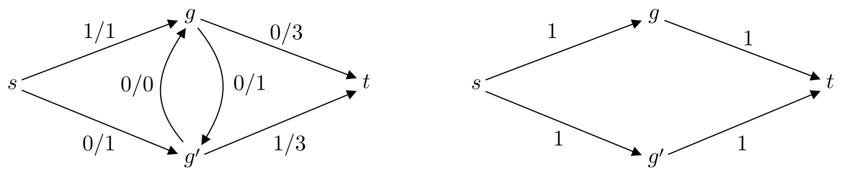

Fig. 5. Example construction in the proof of Theorem 3.13, in the case of free disposal case.

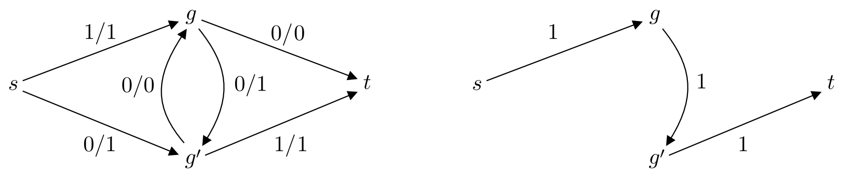

Fig. 6. Example construction in the proof of Theorem 3.13, in the case of lack free disposal case.

actually immaterial. Thus, instances over OR-bids will be next equivalently viewed as instances over atomic bids. Moreover, recall that, by assuming that each bid is defined over singleton sets of transformations, we can get rid of the type of the bids and of their associated payments. Thus, as in Section 3.1, we shall analyze a setting where a MMUCA instance $\mathcal{A}$ is just defined as a tuple $\langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$, where $\mathcal{T}$ is a multi-set of transformations, and where a solution is a sequence of transformations taken from $\mathcal{T}$ and satisfying condition (3) in Definition 2.1.

We start by considering the case where every transformation requires and produces an item of one good at most. We show that Feasibility is feasible in polynomial time both under the free disposal assumption and without free disposal.

Theorem 3.13. *On the class* $\mathcal{C}_{\mathrm{OR}}(1, 1, 1, 1, \infty, \infty)$ *restricted to instances defined over singleton sets of transformations,* Feasibility *is in* P.

Proof. Consider first the case of free disposal. Let $\mathcal{A} = \langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ be a MMUCA such that $\mathcal{A} \in \mathcal{C}_{\mathrm{OR}}(1, 1, 1, 1, \infty, \infty)$. Based on $\mathcal{A}$, we view the goods in G as nodes and we build a directed graph (N, E) such that $N = G \cup \{s, t\}$ and $E = \{(g, g') \mid \{g, g'\} \subseteq G \wedge g \neq g'\} \cup \{(s, g), (g, t) \mid g \in G\}$, where s and t do not occur in G.

The idea is to associate with (N, E) the *flow network* $\langle (N, E), u, \ell\rangle$, $u : E \rightarrow \mathbb{R}^{\geqslant 0}$, $\ell : E \rightarrow \mathbb{R}^{\geqslant 0}$, where each edge $e \in E$ has a capacity $u(e)$ and a lower bound $\ell(e)$ that bound, respectively, the maximum and minimum quantity of flow that can cross it, and where the node s (resp., t) is the source (resp., the sink). Recall first, e.g., from [22], that a *flow assignment* on $\langle (N, E), u, \ell\rangle$ is a function $f : E \rightarrow \mathbb{R}^{\geqslant 0}$ satisfying the following constraints: $\ell(e) \leqslant f(e) \leqslant u(e)$, for each $e \in E$ (*capacity constraints*), and $\sum_{w \in N} f(w, v) - \sum_{r \in N} f(v, r) = 0$, for each $v \in N \setminus \{s, t\}$ (*conservation constraints*). Moreover, recall that the *value* of f is the number $\sum_{w \in N} f(w, t)$. Then, we define edge capacities and lower bounds as follows:

- For each edge of the form (s, g), let $u(s, g) = \mathcal{U}_{in}(g) + |\{\langle \emptyset, \mathcal{O}\rangle \in \mathcal{T} \mid \mathcal{O}(g) > 0\}|$ and $\ell(s, g) = \mathcal{U}_{in}(g)$. That is, the sink s pushes in the network via the edge (s, g) at least $\mathcal{U}_{in}(g)$ units of flow, plus at most $|\{\langle \emptyset, \mathcal{O}\rangle \in \mathcal{T} \mid \mathcal{O}(g) > 0\}|$ additional units associated with transformations producing g and requiring no goods as input.
- For each edge of the form (g, t), let $u(g, t) = \mathcal{U}_{out}(g) + M(g)$ and $\ell(g, t) = \mathcal{U}_{out}(g)$. That is, the source t absorbs via the edge (g, t) at least $\mathcal{U}_{out}(g)$ units of flow, plus at most $M(g)$ additional units. In particular, $M(g)$ has to be set as an upper bound on the maximum number of goods of type g being produced over all the possible solutions to $\mathcal{A}$. For instance, we can set $M(g) = \sum_{g \in G} \mathcal{U}_{in}(g) + |\mathcal{T}|$.
- For each edge of the form (g, g'), let $u(g, g') = |\langle \mathcal{I}, \mathcal{O}\rangle \in T \mid \mathcal{I}(g) > 0 \wedge \mathcal{O}(g') > 0|$ and $\ell(g, g') = 0$. That is, this edge is prepared to route a number of units of flow corresponding to the number of transformations receiving g as input and producing g' as output.

To exemplify the above construction, consider the very simple MMUCA instance $\bar{\mathcal{A}} = \langle \bar{G}, \bar{\mathcal{T}}, \bar{\mathcal{U}}_{in}, \bar{\mathcal{U}}_{out}\rangle$ such that $\bar{G} = \{g, g'\}$, $\bar{T} = \{\langle \{g\}, \{g'\}\rangle, \langle \{\}, \{g'\}\rangle\}$, $\bar{\mathcal{U}}_{in}(g) = 1$, $\bar{\mathcal{U}}_{in}(g') = 0$, $\bar{\mathcal{U}}_{out}(g) = 0$, and $\bar{\mathcal{U}}_{out}(g') = 1$. The associated flow network is reported in Fig. 5, together with a graphical illustration of an admissible flow having value 2 (where edges associated with an empty flow are omitted). Note that such admissible flow corresponds to a solution to $\bar{A}$, where the transformation $\langle \{\}, \{g'\}\rangle$ is used to produce a unit of g' so that the auctioneer ends up with a unit of g' plus the unit of g initially available to her. Another admissible flow, but having value 1, is reported in the right part of Fig. 6. There, the flow corresponds to a solution where the good g initially available to the auctioneer is transformed via $\langle \{g\}, \{g'\}\rangle$ into the good g', which is the desired outcome. In fact, the above correspondences are not by chance.

Indeed, we claim that *there is a solution to $\mathcal{A}$ under the free disposal assumption $\Leftrightarrow$ there is a flow assignment on $\langle (N,E),u,\ell\rangle$*.

($\Rightarrow$) Assume that σ is a solution to $\mathcal{A}$, and let $\mathcal{T}_\sigma$ be the multi-set of all the transformations in σ. Then, consider the function $f: E \rightarrow \mathbb{R}^{\geqslant 0}$ defined as follows. For each edge of the form (s,g), let $f_\sigma(s,g) = \mathcal{U}_{in}(g) + |\{\langle \emptyset, \mathcal{O}\rangle \in \mathcal{T}_\sigma \mid \mathcal{O}(g) > 0\}|$. For each edge of the form (g,g'), let $f_\sigma(g,g') = |\langle \mathcal{I},\mathcal{O}\rangle \in T_\sigma \mid \mathcal{I}(g) > 0 \wedge \mathcal{O}(g') > 0|$. And, for each edge of the form (g,t), let $f_\sigma(g,t) = f_\sigma(s,g) + \sum_{(g',g)\in E \wedge g' \in G} f_\sigma(g,g') - \sum_{(g,g')\in E\wedge g'\in G} f_\sigma(g,g')$. Note that, by construction, the function f_σ satisfies the conservation constraints of the flow network $\langle (N,E),u,\ell\rangle$, and the capacity constraints for the edges of the form (s,g), (g,g'), and (g',g) with $g' \in G$. Consider then any edge of the form (g,t) and observe that since σ is a solution, it must be the case that $f_\sigma(g,t) \geqslant \mathcal{U}_{out}(g)$. That is, f_σ also satisfies the capacity constraints for the edges of the form (g,t) and, hence, f_σ is a flow assignment on $\langle (N,E),u,\ell\rangle$.

($\Leftarrow$) Assume that $f: E \rightarrow \mathbb{R}^{\geqslant 0}$ is a flow assignment on $\langle (N,E),u,\ell\rangle$, and let V be its value. In particular, since all edge capacities and lower bounds are integral, we can assume, w.l.o.g., that f is integral as well (see, e.g., [22]). That is, $f(e) \in \mathbb{N}$, for each $e \in E$. Based on f, we now build a succession of transformations $\sigma_{in}, \sigma_1, \dots, \sigma_V$, where σ_{in} and σ_i, for each $i \in \{1,\dots,V\}$, are subsequences defined as follows:

- For each good $g \in G$, σ_{in} exactly contains $f(s,g)$–$\mathcal{U}_{in}(g)$ transformations $\langle \emptyset, \mathcal{O}\rangle \in \mathcal{T}$ such that $\mathcal{O}(g) > 0$. In particular, the relative order of application of these transformations is immaterial.
- Note that, since the value of f is V and since f is integral, we are guaranteed that there are V (not necessarily distinct) paths, i.e., succession of edges, $\pi_1,\dots,\pi_V$ from s to t such that for each edge $e \in E$, $\ell(e) \leqslant |\{\pi_i \mid e$ occurs in $\pi_i\}| \leqslant u(e)$. Indeed, each path π_i serves the purpose of routing one unit of flow from s to t. In particular, for each $i \in \{1,\dots,V\}$, observe that π_i is of the form $(s,g), \bar{\pi}_i, (g',t)$, where g and g' are two goods in G, and where $\bar{\pi}_i$ is a (possibly empty, if $g = g'$) path defined over the goods in G only. The idea is to define σ_i as the succession of the transformations that are naturally associated with $\bar{\pi}_i$, and which is in fact obtained by replacing each edge of the form $(\bar{g},\bar{g}')$ occurring in $\bar{\pi}_i$ with a transformation $\langle \mathcal{I},\mathcal{O}\rangle \in \mathcal{T}$ such that $\mathcal{I}(\bar{g}) > 0$ and $\mathcal{O}(\bar{g}') > 0$.

Observe now that σ_{in} can be actually executed, and after it, the auctioneer holds $f(s,g)$ units of good g, for each $g \in G$. Moreover, for each $i \in \{1,\dots,V\}$, the succession σ_i implements a (possibly empty) sequence of transformations for one of these goods. After σ_i, the good initially owned by the auctioneer after σ_{in} is then possibly transformed in another good and no further transformation is applied on it. In particular, note that all the transformations of the form σ_i can be actually executed, since for each edge of the form (g,g'), $l(g,g') \leqslant |\{\pi_i \mid (g,g')$ occurs in $\pi_i\}| \leqslant u(g,g') = |\langle \mathcal{I},\mathcal{O}\rangle \in T \mid \mathcal{I}(g) > 0 \wedge \mathcal{O}(g') > 0|$. Moreover, after that σ_V is executed, the auctioneer holds $f(g,t)$ units of good g, for each $g \in G$. Since, $f(g,t) \geqslant \ell(g,t) = \mathcal{U}_{out}(g)$ holds, we have that $\sigma_{in}, \sigma_1, \dots, \sigma_V$ is a solution to $\mathcal{A}$.

After the claim above, the result in the case of free disposal easily follows, as the existence of a flow assignment on a flow network can be decided in polynomial time (see [22]).

In order to conclude the proof, we observe that the case of lack of free disposal can be faced with a simple adaptation of the above construction. Indeed, we have just to modify the upper bound $M(g)$, by setting it to the value $|\langle \mathcal{I},\emptyset\rangle \in \mathcal{T} \mid \mathcal{I}(g) > 0|$, and thereby guaranteeing that any surplus of good g can be produced only if there is some transformation that is prepared to consume it. Then, the result can be established by following precisely the same line of reasoning as in the proof of the claim above, and by noticing that the novel upper bound guarantees that the auctioneer ends up exactly with the desired $\mathcal{U}_{out}(g)$ units of good g, after that up to $|\langle \mathcal{I},\emptyset\rangle \in \mathcal{T} \mid \mathcal{I}(g) > 0|$ surplus units of g are consumed by transformations producing as output no goods. As an example, Fig. 6 shows the modified flow network for the instance $\bar{A}$ and an admissible solution where the auctioneer ends up exactly with one unit of g'. Note that the flow depicted in the left part of Fig. 5 is not admissible w.r.t. the modified network. □

Let us now turn to the scenario where every type of good is consumed by one transformation at most, i.e., let us assume that output degrees are unitary at most.

Theorem 3.14. *On the class $\mathcal{C}_{\mathrm{OR}}(\infty,\infty,\infty,\infty,\infty,1)$ restricted to instances defined over singleton sets of transformations,* FEASIBILITY *is in* P.

Proof. Let $\mathcal{A} = \langle G,\mathcal{T},\mathcal{U}_{in},\mathcal{U}_{out}\rangle$ be a MMUCA such that $\mathcal{A} \in \mathcal{C}_{\mathrm{OR}}(\infty,\infty,\infty,\infty,\infty,1)$. Let $\mathcal{U}_G$ be the multi-set of goods the auctioneer owns (initially, $\mathcal{U}_G = \mathcal{U}_{in}$). Let $\mathcal{T}_a \subseteq \mathcal{T}$ be the multi-set of transformations that are *active* (initially, $\mathcal{T}_a = \{\langle \mathcal{I},\mathcal{O}\rangle \in \mathcal{T} \mid \mathcal{U}_{in}(g) \geqslant \mathcal{I}(g), \forall g \in G\}$). Consider the approach where we arbitrarily pick and execute a transformation $\langle \mathcal{I},\mathcal{O}\rangle \in \mathcal{T}_a$, until the set $\mathcal{T}_a$ is empty, and update the variables as follows:

- $\mathcal{U}_G(g) := \mathcal{U}_G(g) - \mathcal{I}(g) + \mathcal{O}(g), \forall g \in G$;
- $\mathcal{T} := \mathcal{T} \setminus \{\langle \mathcal{I},\mathcal{O}\rangle\}$;
- $\mathcal{T}_a := \{\langle \mathcal{I}',\mathcal{O}'\rangle \in \mathcal{T} \mid \mathcal{U}_G(g) \geqslant \mathcal{I}'(g), \forall g \in G\}$.

Note that the approach takes polynomially many steps at most, each one feasible in polynomial time. Then, we claim that *there is a solution to $\mathcal{A}$ under the free disposal assumption $\Leftrightarrow$ when $\mathcal{T}_a$ is empty, $\mathcal{U}_G(g) \geqslant \mathcal{U}_{out}(g), \forall g \in G$.*

($\Leftarrow$) Assume that, after the final step, $\mathcal{U}_G(g) \geqslant \mathcal{U}_{out}(g), \forall g \in G$. By construction, the sequence of transformations picked as discussed above is a solution to $\mathcal{A}$.

($\Rightarrow$) Assume that $\mathcal{A}$ admits a solution $\hat{\sigma}$, and let σ be the sequence of transformations picked as discussed above. Note that the execution of any transformation $\langle \mathcal{I}, \mathcal{O} \rangle \in \mathcal{T}_a$ does not invalidate the possibility of executing other transformations in $\mathcal{T}_a$, because output degrees are unitary (at most). Thus, each transformation in $\hat{\sigma}$ also belongs to σ, and the order of their execution is immaterial. Since the goods the auctioneer wants to end up with are not required in input by any transformation (because the output degree is unitary), the fact that σ might contain more transformations than $\hat{\sigma}$ is irrelevant.

In the case of lack of free disposal, it is easily seen that the only difference is that, when the algorithm stops, we must check whether $\mathcal{U}_G(g) = \mathcal{U}_{out}(g), \forall g \in G$. □

In the case where every type of good is produced by one transformation at most, tractability can be established under the free disposal assumption.

Theorem 3.15. *On the class* $\mathcal{C}_{\text{OR}}(\infty, \infty, \infty, \infty, 1, \infty)$ *restricted to instances defined over singleton sets of transformations,* FEASIBILITY *is in* P *under the free disposal assumption.*

Proof. Let $\mathcal{A} = \langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$ be a MMUCA such that $\mathcal{A} \in \mathcal{C}_{\text{OR}}(\infty, \infty, \infty, \infty, 1, \infty)$.

A transformation $\langle \mathcal{I}, \mathcal{O} \rangle \in \mathcal{T}$ is said *necessary* w.r.t. a good $g \in G$ if: (a) $\mathcal{O}(g) > 0$; or (b) there is a transformation $\langle \mathcal{I}', \mathcal{O}' \rangle \in \mathcal{T}$ such that $\mathcal{O}'(g) > 0$, and $\langle \mathcal{I}, \mathcal{O} \rangle$ is necessary w.r.t. some good g' such that $\mathcal{I}'(g') > 0$. Let $\mathcal{T}_n \subseteq \mathcal{T}$ denote the multi-set of all the transformations that are necessary w.r.t. some good g with $\mathcal{U}_{out}(g) > 0$.

Moreover, as in the proof of Theorem 3.14, let $\mathcal{U}_G$ be the multi-set of goods the auctioneer owns (initially, $\mathcal{U}_G = \mathcal{U}_{in}$), and let $\mathcal{T}_a \subseteq \mathcal{T}$ be the set of transformations that are *active* (initially, $\mathcal{T}_a = \{\langle \mathcal{I}, \mathcal{O} \rangle \in \mathcal{T} \mid \mathcal{U}_{in}(g) \geqslant \mathcal{I}(g), \forall g \in G\}$).

We define an algorithm that arbitrarily picks and executes an active transformation $\langle \mathcal{I}, \mathcal{O} \rangle \in \mathcal{T}_a \cap \mathcal{T}_n$, until $\mathcal{T}_n \cap \mathcal{T}_a = \emptyset$, by updating the variables as follows:

- $\mathcal{U}_G(g) := \mathcal{U}_G(g) - \mathcal{I}(g) + \mathcal{O}(g), \forall g \in G$;
- $\mathcal{T}_n := \mathcal{T}_n \setminus \{\langle \mathcal{I}, \mathcal{O} \rangle\}$;
- $\mathcal{T} := \mathcal{T} \setminus \{\langle \mathcal{I}, \mathcal{O} \rangle\}$;
- $\mathcal{T}_a := \{\langle \mathcal{I}', \mathcal{O}' \rangle \in \mathcal{T} \mid \mathcal{U}_G(g) \geqslant \mathcal{I}(g), \forall g \in G\}$.

We now claim that *there is a solution to $\mathcal{A}$ under the free disposal assumption $\Leftrightarrow$ the algorithm stops because $\mathcal{T}_n = \emptyset$, and in the final step $\mathcal{U}_G(g) \geqslant \mathcal{U}_{out}(g), \forall g \in G$.*

($\Leftarrow$) Assume that the algorithm stops because $\mathcal{T}_n = \emptyset$. Assume that in the final step $\mathcal{U}_G(g) \geqslant \mathcal{U}_{out}(g), \forall g \in G$. Then, by construction, the sequence of transformations picked as discussed above is a solution to $\mathcal{A}$.

($\Rightarrow$) Suppose that $\mathcal{A}$ admits a solution. Observe that, at each step, the selection of the transformation that has to be executed, among all the active transformations in $\mathcal{T}_n$, is immaterial, since every type of good is produced by one transformation at most and since all the transformations in $\mathcal{T}_n$ are necessary for producing the goods required by the auctioneer, as there is no transformation producing the types of goods initially owned by the auctioneer (because the input degree is unitary). Thus, all transformations in $\mathcal{T}_n$ must belong to any solution sequence. In fact, a solution to $\mathcal{A}$ might contain some further transformations that are not necessary for satisfying the requirements of the auctioneer. Yet, if $\mathcal{A}$ admits a solution, then it admits a solution containing no unnecessary transformations, which will be hence detected by the algorithm.

To conclude, just notice that the algorithm is feasible in polynomial time. □

Under the lack of free disposal, the class $\mathcal{C}_{\text{OR}}(\infty, \infty, \infty, \infty, 1, \infty)$ is not an island of tractability for FEASIBILITY, as we have already observed in Theorem 3.9 and in Theorem 3.10. An interesting open question is whether FEASIBILITY is tractable on the class $\mathcal{C}_{\text{OR}}(1, 1, k, k, 1, k)$, for some fixed constant k—see, again, Fig. 2.

3.4. Tractable instances for XOR*-bids over singleton sets of transformations*

We now conclude the analysis of the scenario where bids are defined over singleton sets of transformations by considering the case of XOR-bids. Firstly, we observe that the proof of Theorem 3.15 can easily be adapted to cope with XOR-bids, based on the fact that executing necessary transformations is mandatory as to solve the instance at hand. Thus, if some necessary transformation cannot be executed because of some given XOR-condition, then no solution exists.

Theorem 3.16. *On the class* $\mathcal{C}_{\text{XOR}}(\infty, \infty, \infty, \infty, 1, \infty)$ *restricted to instances defined over singleton sets of transformations,* FEASIBILITY *is in* P *under the free disposal assumption.*

Precisely as in the case of OR-bids, under the lack of free disposal the above class is no longer an island of tractability, while the tractability on $\mathcal{C}_{\text{XOR}}(1, 1, k, k, 1, k)$, for some fixed constant k, is open. However, tractability can be established by focusing on unitary output degree, as shown next.

Theorem 3.17. *On the class $\mathcal{C}_{\text{XOR}}(\infty, \infty, \infty, \infty, 1, 1)$ restricted to instances defined over singleton sets of transformations,* Feasibility *is in* P *under the lack of free disposal assumption.*

Proof. Let $\mathcal{A} = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ be a MMUCA instance that belongs to the class $\mathcal{C}_{\text{XOR}}(\infty, \infty, \infty, \infty, 1, 1)$. Let $\mathcal{T} = \bigcup_{\ell=1}^{n} bids(\mathfrak{L}_\ell)$ denote the multi-set of all transformations occurring in $\mathcal{A}$—recall that, given our interest in Feasibility and in the setting where atomic bids are restricted over singleton sets of transformations, any XOR-bid can be viewed as an expression t_1 XOR t_2 XOR ... XOR t_k, where t_i is a transformation (rather than an atomic bid), for each $1 \leqslant i \leqslant k$. Let $\mathcal{T}_f \subseteq \mathcal{T}$ be the multi-set of all the transformations that are not implementable due to some XOR-bid (initially, $\mathcal{T}_f = \emptyset$). Moreover, let $\mathcal{U}_G$, $\mathcal{T}_a \subseteq \mathcal{T}$, and $\mathcal{T}_n \subseteq \mathcal{T}$ be the multi-sets as in the proof of Theorem 3.15.

Observe that, since $\mathcal{A} \in \mathcal{C}_{\text{XOR}}(\infty, \infty, \infty, \infty, 1, 1)$ holds, any transformation $\langle \mathcal{I}, \mathcal{O}\rangle \in \mathcal{T}$ exactly occurs in one XOR-bid in $\{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$, which we denote by $\text{xor}(\langle \mathcal{I}, \mathcal{O}\rangle)$.

Consider the following two-steps approach. In the first step, we arbitrarily pick and execute an active transformation $\langle \mathcal{I}, \mathcal{O}\rangle \in (\mathcal{T}_a \cap \mathcal{T}_n) \setminus \mathcal{T}_f$ that is not forbidden by the XOR-bids, until $(\mathcal{T}_a \cap \mathcal{T}_n) \setminus \mathcal{T}_f = \emptyset$, by updating the variables as follows:

- $\mathcal{U}_G(g) := \mathcal{U}_G(g) - \mathcal{I}(g) + \mathcal{O}(g), \forall g \in G$;
- $\mathcal{T}_n := \mathcal{T}_n \setminus \{\langle \mathcal{I}, \mathcal{O}\rangle\}$;
- $\mathcal{T}_f = \mathcal{T}_f \cup bids(\text{xor}(\langle \mathcal{I}, \mathcal{O}\rangle))$;
- $\mathcal{T} := \mathcal{T} \setminus \{\langle \mathcal{I}, \mathcal{O}\rangle\}$;
- $\mathcal{T}_a := \{\langle \mathcal{I}', \mathcal{O}'\rangle \in \mathcal{T} \mid \mathcal{U}_G(g) \geqslant \mathcal{I}(g), \forall g \in G\}$.

In the second step, we consume all surplus goods as long as it is possible. Hereinafter, we assume that $\mathcal{T}$, $\mathcal{U}_G$, and $\mathcal{T}_f$ are those computed in the last iteration of the first step. Let $\mathcal{T}_i \subseteq \mathcal{T}$ be the multi-set of all transformations that are implementable starting from the surplus goods (initially, $\mathcal{T}_i = \{\langle \mathcal{I}, \mathcal{O}\rangle \in \mathcal{T} \mid (\mathcal{U}_G(g) – \mathcal{U}_{out}(g)) \geqslant \mathcal{I}(g)\}$). Then, we arbitrarily pick and execute an implementable transformation $\langle \mathcal{I}, \mathcal{O}\rangle \in \mathcal{T}_i \setminus \mathcal{T}_f$ that is not forbidden by the XOR-bids, until $T_i \setminus \mathcal{T}_f = \emptyset$, by updating all the variables as follows:

- $\mathcal{U}_G(g) := \mathcal{U}_G - \mathcal{I}(g) + \mathcal{O}(g), \forall g \in G$;
- $\mathcal{T} := \mathcal{T} \setminus \{\langle \mathcal{I}, \mathcal{O}\rangle\}$;
- $\mathcal{T}_i := \{\langle \mathcal{I}', \mathcal{O}'\rangle \in \mathcal{T} \mid (\mathcal{U}_G(g) – \mathcal{U}_{out}(g)) \geqslant \mathcal{I}'(g), \forall g \in G\}$;
- $\mathcal{T}_f = \mathcal{T}_f \cup bids(\text{xor}(\langle \mathcal{I}, \mathcal{O}\rangle))$.

We now claim that *there is a solution to $\mathcal{A}$ under the lack of free disposal $\Leftrightarrow$ the algorithm stops with $\mathcal{T}_n = \mathcal{T}_i = \emptyset$ and $\mathcal{U}_G(g) = \mathcal{U}_{out}(g), \forall g \in G$.*

($\Leftarrow$) Assume that, when the algorithm ends, $\mathcal{T}_n = \mathcal{T}_i = \emptyset$ and $\mathcal{U}_G(g) = \mathcal{U}_{out}(g), \forall g \in G$. Then, by construction, the sequence of transformations picked as discussed above is a solution to $\mathcal{A}$.

($\Rightarrow$) Suppose that $\mathcal{A}$ admits a solution. Observe that, at each iteration of the first step, the selection of the transformation that has to be executed, among all the active transformations in $\mathcal{T}_n$ is arbitrary, since each type of good is produced by one transformation at most and since all the transformations in $\mathcal{T}_n$ are necessary for producing the goods required by the auctioneer. Moreover, the order in which the transformations that belong to $\mathcal{T}_i$ are executed is immaterial since each type of good is consumed by one transformation at most and all the transformations in $\mathcal{T}_i$ are necessary to consume the surplus goods obtained by implementing the transformations in $\mathcal{T}_n$. Thus, all transformations in $\mathcal{T}_n$ and in $\mathcal{T}_i$ must belong to any solution sequence. Indeed, if some transformation in $\mathcal{T}_n$ or $\mathcal{T}_i$ cannot be executed due to some XOR-bids, some required good cannot be produced or some surplus good cannot be consumed. In fact, note that a solution to $\mathcal{A}$ might contain some further transformations that are not necessary for producing the required goods and, thus, some additional transformations for consuming the surplus goods possibly produced by these unnecessary transformations. Anyway, if $\mathcal{A}$ admits a solution, then it admits a solution containing no unnecessary transformations, which will be hence detected by the algorithm.

To conclude, note that the algorithm takes polynomial time. □

3.5. Feasibility *for arbitrary sets of transformations*

In this section, we complete the picture of the complexity of Feasibility, by focusing on the general scenario where bids can be defined over arbitrary sets of transformations. Of course, all hardness results provided for singleton sets of transformations still hold in this more general setting. Thus, in the following, we shall just illustrate NP-hardness results that are more stringent than those in Fig. 2, in addition to isolating novel islands of tractability. Most proofs are based on straightforwardly adapting, or even just inspecting, those illustrated above, and hence details are omitted.

Concerning the hardness results, we observe that any transformation of the form $\langle \mathcal{I}, \mathcal{O} \rangle$ with $|\{g \mid g \in G, \mathcal{I}(g) > 0\}| > 0$ (i.e., with input variability greater than 1), can be equivalently reformulated in terms of an atomic bid $\langle \mathcal{B}, p, \text{TYPE} \rangle$ with $\text{TYPE} = \text{FULL}$ and $\mathcal{B} = \{\langle \{\}, \mathcal{O} \rangle\} \cup \{\langle \{g\}, \{\} \rangle \mid \mathcal{I}(g) > 0\}$. Indeed, $\mathcal{B}$ precisely prescribe to produce all the goods in $\mathcal{O}$ and to consume each good g in $\mathcal{I}$. In fact, this transformation does not affect any parameter of the game, but the input variability that is now guaranteed to be unitary. Hence, we can strengthen Theorem 3.5, Theorem 3.9, and Theorem 3.12.

Theorem 3.18. Feasibility *is* NP-*hard under the free disposal assumption, even when restricted to the class* $\mathcal{C}(1, 1, 1, 1, 2, 2)$. *Moreover, it is* NP-*hard under the lack of free disposal, even when restricted to the class* $\mathcal{C}(1, 1, 1, 1, 1, 2)$.

Concerning tractability results, observe the proofs of Theorem 3.15 and Theorem 3.16 can be easily adapted to deal with bids over arbitrary sets of transformations, by defining a transformation as *necessary* if it occurs in some bid with $\text{TYPE} = \text{FULL}$ together with a transformation that is in its turn necessary (and, of course, by focusing on the smallest set of necessary transformations, given the circularity of the definition). With this modification in place, the following result derives by inspecting such proofs.

Theorem 3.19. *On the classes* $\mathcal{C}_{\text{OR}}(\infty, \infty, \infty, \infty, 1, \infty)$ *and* $\mathcal{C}_{\text{XOR}}(\infty, \infty, \infty, \infty, 1, \infty)$, Feasibility *is in* P *under the free disposal assumption.*

Let us now conclude our analysis by extending to the case of arbitrary sets of transformations the two approaches discussed in the proofs of Theorem 3.14 and Theorem 3.17, respectively.

Theorem 3.20. *On the class* $\mathcal{C}_{\text{OR}}(\infty, \infty, \infty, \infty, \infty, 1)$, Feasibility *is in* P.

Proof (*Sketch*). Let $\mathcal{A}$ be a MMUCA instance in $\mathcal{C}_{\text{OR}}(\infty, \infty, \infty, \infty, \infty, 1)$. The algorithm for deciding the Feasibility of $\mathcal{A}$ works in several subsequent iterations. At each iteration, active transformations are executed, until one exists. In particular, we avoid the execution of those transformations that have been executed in some previous iterations and that violated some bid constraints (indicated in the following as *forbidden* transformations).

Formally, let $\mathcal{U}_G$ be the multi-set of goods the auctioneer owns (initially, $\mathcal{U}_G = \mathcal{U}_{in}$), let $\mathcal{T}_a \subseteq \mathcal{T}$ be the multi-set of transformations that are *active* (initially, $\mathcal{T}_a = \{\langle \mathcal{I}, \mathcal{O} \rangle \in \mathcal{T} \mid \mathcal{U}_{in}(g) \geqslant \mathcal{I}(g), \forall g \in G\}$), and let $\mathcal{T}_f \subset \mathcal{T}$ be the multi-set of *forbidden* transformations (initially, $\mathcal{T}_f = \emptyset$). Then, each iteration consists in arbitrarily picking and executing a transformation $\langle \mathcal{I}, \mathcal{O} \rangle \in \mathcal{T}_a \setminus \mathcal{T}_f$, until the set $\mathcal{T}_a \setminus \mathcal{T}_f$ is empty, and updating the variables as follows:

- $\mathcal{U}_G(g) := \mathcal{U}_G(g) - \mathcal{I}(g) + \mathcal{O}(g), \forall g \in G$;
- $\mathcal{T} := \mathcal{T} \setminus \{\langle \mathcal{I}, \mathcal{O} \rangle\}$;
- $\mathcal{T}_a := \{\langle \mathcal{I}', \mathcal{O}' \rangle \in \mathcal{T} \mid \mathcal{U}_G(g) \geqslant \mathcal{I}'(g), \forall g \in G\}$.

At the end of the iteration, if no bid constraint has been violated (i.e., for each bid $\langle \mathcal{B}, p, \text{TYPE} \rangle$ with $\text{TYPE} = \text{FULL}$, all transformations in $\mathcal{B}$ have been executed), the algorithm stops and the requirements of the auctioneer are checked (i.e., if $\mathcal{U}_G(g) \geqslant \mathcal{U}_{out}(g), \forall g \in G$). Otherwise, $\mathcal{T}_f$ is updated by adding all transformations that have been implemented in the last iteration and that occur in some bid with $\text{TYPE} = \text{FULL}$ together with a transformation that has not been executed. In this latter case, the algorithm is reiterated, by initializing again $\mathcal{U}_G$, $\mathcal{T}_a$, and $\mathcal{T}$.

By exploiting the line of reasoning in the proof of Theorem 3.14, in the light of the fact that the order of execution of activities is $\mathcal{T}_a$ is immaterial and that transformations in $\mathcal{T}_f$ cannot occur in any solution, we can conclude that *there is a solution to* $\mathcal{A}$ *under the free disposal assumption* $\Leftrightarrow$ *when the algorithm stops* $\mathcal{U}_G(g) \geqslant \mathcal{U}_{out}(g), \forall g \in G$. The same line of reasoning can then be extended to the class $\mathcal{C}_{\text{OR}}(\infty, \infty, \infty, \infty, \infty, 1)$ in case of lack of free disposal with the only difference that, when the algorithm stops, we must check whether $\mathcal{U}_G(g) = \mathcal{U}_{out}(g), \forall g \in G$. □

Theorem 3.21. *On the class* $\mathcal{C}_{\text{XOR}}(\infty, \infty, \infty, \infty, 1, 1)$, Feasibility *is in* P *under the lack of free disposal assumption.*

Proof. [Proof Sketch] To establish the result we can follow the same line of reasoning as in the two-steps algorithm exploited in the proof of Theorem 3.17, provided the following adaptations. In the first step, we must adopt the definition of *necessary* transformations exploited for establishing Theorem 3.19. And, in the second step, we must consider a multi-set $\mathcal{T}_{req}$ of required transformations with the same approach as in the proof of Theorem 3.20, by eventually checking whether all of them can be executed. □

4. Graph-based structural restrictions do not help

Many NP-hard problems in different application areas, ranging, e.g., from Constraint Satisfaction to Database Theory, are known to be efficiently solvable when restricted to instances that can be modeled via (nearly)acyclic graphs. Indeed, on

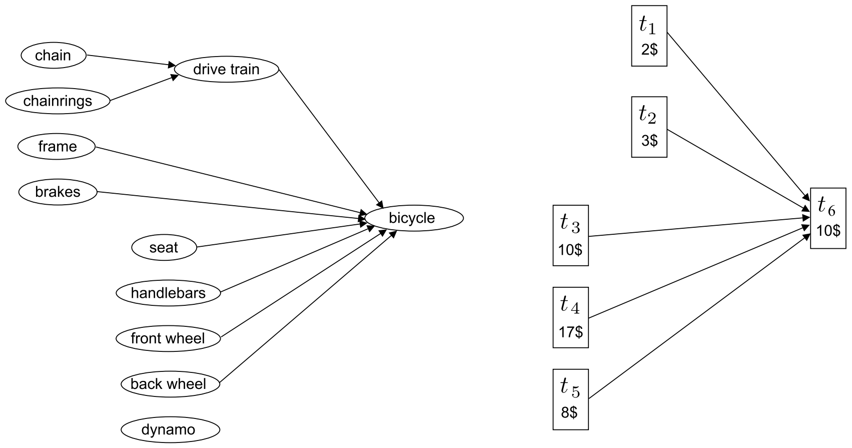

Fig. 7. Transformations and goods graphs for the MMUCA instance in Example 1.1.

these kinds of instances, solutions can usually be computed via dynamic programming, by incrementally processing the acyclic structure, according to some of its topological orderings. Therefore, one may naturally expect that these structural restrictions are also beneficial to isolate tractable MMUCAs. The question is investigated in the following.

Let $\mathcal{A}$ be a MMUCA instance over a set G of goods, and let $\mathcal{T}_r$ denote the multi-set of all transformations occurring in its bids. Interactions in $\mathcal{A}$ are modeled by considering two different graph structures:

- The *transformations graph* of $\mathcal{A}$, denoted by $\mathsf{TG}(\mathcal{A})$, is a directed graph (N_t, E_t) where nodes are in one-to-one correspondence with transformations in $\mathcal{T}_r$ (i.e., $t_h \in N_t$ if and only if $t_h = \langle \mathcal{I}_h, \mathcal{O}_h, p_h\rangle \in \mathcal{T}_r$), and where there is an edge $(t_i, t_j) \in E_t$ from $t_i = \langle \mathcal{I}_i, \mathcal{O}_i, p_i\rangle$ to $t_j = \langle \mathcal{I}_j, \mathcal{O}_j, p_j\rangle$ if there exists a good $g \in G$ such that $\mathcal{I}_j(g) > 0$ and $\mathcal{O}_i(g) > 0$. The undirected version of $\mathsf{TG}(\mathcal{A})$ is denoted by $\overline{\mathsf{TG}}(\mathcal{A})$.
- The *goods graph* of $\mathcal{A}$, denoted by $\mathsf{GG}(\mathcal{A})$, is a directed graph (N_g, E_g) where nodes are in one-to-one correspondence with the goods in G, and where there is an edge $(g, g') \in E_g$ from g to g' if there exists a transformation $\langle \mathcal{I}_h, \mathcal{O}_h, p_h\rangle \in \mathcal{T}_r$ such that $\mathcal{I}_h(g) > 0$ and $\mathcal{O}_h(g') > 0$. The undirected version of $\mathsf{GG}(\mathcal{A})$ is denoted by $\overline{\mathsf{GG}}(\mathcal{A})$.

Example 4.1. Consider again the setting discussed in Example 1.1 (then, formalized as the MMUCA instance $\bar{\mathcal{A}}$ in Example 2.2) and the supply chain graphically illustrated in Fig. 1. The transformations graph $\mathsf{TG}(\bar{\mathcal{A}})$ and the goods graph $\mathsf{GG}(\bar{\mathcal{A}})$ are graphically reported on the right and on the left of Fig. 7, respectively. Note that both graphs, as well as their undirected versions, are acyclic. $\lhd$

Our first result is very bad news on directed and undirected transformations graphs, and on directed goods graphs. The result holds under the free disposal assumption and under the lack of free disposal.

Theorem 4.2. Feasibility *is* NP-*complete, even when restricted to the classes* $\{\mathcal{A} \mid \overline{\mathsf{TG}}(\mathcal{A})$ *is acyclic*$\}$ (*and, hence,* $\{\mathcal{A} \mid \mathsf{TG}(\mathcal{A})$ *is acyclic*$\}$) *and* $\{\mathcal{A} \mid \mathsf{GG}(\mathcal{A})$ *is acyclic*$\}$*, and under atomic bids defined over singleton sets of transformations.*

Proof. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula over the variables $X_1, \dots, X_n$ satisfying the conditions in Lemma 3.2, and let $\mathcal{A}(\Phi) = \langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ be the MMUCA instance over atomic bids (defined over singleton sets of transformations) such that: $G = \bigcup_{i=1}^{n}\{X_i, X_i^T, X_i^F\} \cup \{c_1, \dots, c_m\} \cup \{c_j^i \mid X_i \text{ occurs in } c_j\}$, $\mathcal{U}_{in} = \{X_1, \dots, X_n\}$, $\mathcal{U}_{out} = \{c_1, \dots, c_m\}$, and $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$. In particular, for each variable X_i occurring positively in the clauses c_α and c_β while occurring negatively in c_γ, the set $\mathcal{T}_i$ consists of the set of transformations $\{\langle\{X_i\}, \{X_i^T\}\rangle, \langle\{X_i\}, \{X_i^F\}\rangle, \langle\{X_i^T\}, \{c_\alpha^i, c_\beta^i\}\rangle, \langle\{X_i^F\}, \{c_\gamma^i\}\rangle, \langle\{c_\alpha^i\}, \{c_\alpha\}\rangle, \langle\{c_\beta^i\}, \{c_\beta\}\rangle$, and $\langle\{c_\gamma^i\}, \{c_\gamma\}\rangle\}$. An illustration of $\mathcal{T}_i$ is reported on the left of Fig. 8, where transformations have been indexed for the sake of readability. Recall that deciding Feasibility over the class of instances of the form $\mathcal{A}(\Phi)$ is NP-hard, under the free disposal assumption (Theorem 3.3) and under the lack of free disposal (Theorem 3.8). We next show that such a class of instances is a subset of the classes $\{\mathcal{A} \mid \overline{\mathsf{TG}}(\mathcal{A}) \text{ is acyclic}\}$ and $\{\mathcal{A} \mid \mathsf{GG}(\mathcal{A}) \text{ is acyclic}\}$.

On the top-right part of Fig. 8, the portion of the transformations graph $\mathsf{TG}(\mathcal{A}(\Phi))$ associated with the transformations in $\mathcal{T}_i$ is reported, which we find useful to denote by $\mathsf{TG}(\mathcal{T}_i)$. Note that no cycle occurs in $\mathsf{TG}(\mathcal{T}_i)$, even when we consider its undirected version. Moreover, consider now two arbitrary transformations $t_i \in \mathcal{T}_i$ and $t_j \in \mathcal{T}_j$, and note that there is no good $g \in G$ produced by t_i and consumed by t_j, or vice versa. In fact, only goods of the form $c_1, \dots, c_m$ can in principle be shared between t_i and t_j, but in any case they will produced by both t_i and t_j and there is no transformation consuming them. It follows that $\mathsf{TG}(\mathcal{A}(\Phi))$ and $\overline{\mathsf{TG}}(\mathcal{A}(\Phi))$ are both acyclic.

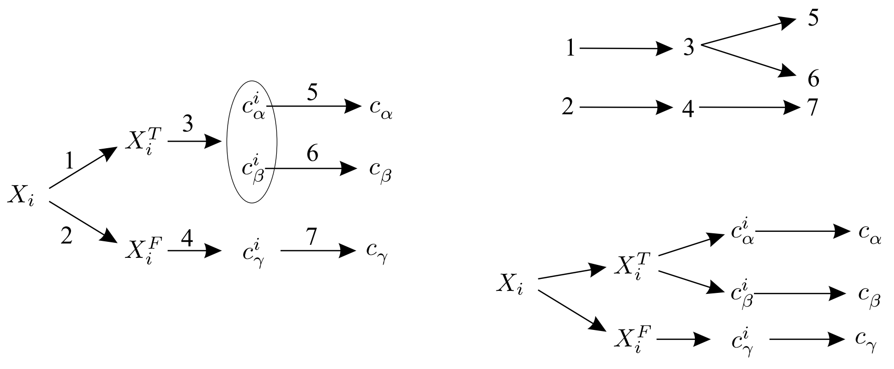

Fig. 8. Transformations and goods graphs in the proof Theorem 4.2.

Finally, on the bottom-right part of Fig. 8, the portion of the good graph $\mathsf{GG}(\mathcal{A}(\Phi))$ associated with the goods manipulated by transformations in $\mathcal{T}_i$ is reported, which we find useful to denote by $\mathsf{GG}(\mathcal{T}_i)$. As observed above, for two arbitrary transformations $t_i \in \mathcal{T}_i$ and $t_j \in \mathcal{T}_j$, $\mathsf{GG}(\mathcal{T}_i)$ and $\mathsf{GG}(\mathcal{T}_j)$ might share goods in the set $\{c_1, \dots, c_m\}$. However, as these goods are produced and never consumed by transformations in $\mathcal{T}$, any node in $\{c_1, \dots, c_m\}$ contains some incoming edge, but no outgoing edges in $\mathsf{GG}(\mathcal{A}(\Phi))$. It follows that $\mathsf{GG}(\mathcal{A}(\Phi))$ is acyclic. However, note that $\overline{\mathsf{GG}}(\mathcal{A}(\Phi))$ is not acyclic in general. □

Note that the above result is interesting in the light that instances with (directed) acyclic goods graphs correspond to transformation processes (cf. [4]). Thus, transformation processes emerge to be as hard as arbitrary trades and exchanges of goods.

We conclude this section by noticing that an NP-hardness result can be obtained even on the remaining class $\{\mathcal{A} \mid \overline{\mathsf{GG}}(\mathcal{A})$ is acyclic$\}$. In this case, differently from the proof of Theorem 4.2, we shall exploit a reduction from a weakly NP-hard problem (as in Theorem 3.10), and it remains open whether a strongly NP-hardness can be obtained.

Again, the result below holds under the free disposal assumption and under the lack of free disposal.

Theorem 4.3. Feasibility *is in* NP*-complete when restricted to the classes* $\{\mathcal{A} \mid \overline{\mathsf{GG}}(\mathcal{A})$ *is acyclic*$\}$ *and under atomic bids defined over singleton sets of transformations.*

Proof. Recall that deciding whether there is a way to partition a bag $S = \{s_1, s_2, \dots, s_n\}$ of integers into two bags S_1 and S_2 such that the sum of the numbers in S_1 equals the sum of the numbers in S_2 is the NP-complete Partition problem [21].

Let $m = \sum_{i=1}^{n} s_i$, and consider the MMUCA instance $\mathcal{A}(S) = \langle G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ such that: $G = \{\alpha, S'\}$, $\mathcal{U}_{in}(\alpha) = m/2$, $\mathcal{U}_{out}(S') = m/2$ and $\mathcal{T} = \bigcup_{i=1}^{n} \{\langle \bigcup_{i=1}^{s_i} \{\alpha\}, \bigcup_{i=1}^{s_i} \{S'\}\rangle\}$. Then, it is immediate to check that there is a solution to $\mathcal{A}(S)$ if, and only if, Partition has a solution on input S. Indeed, each solution to $\mathcal{A}(S)$ is in one-to-one correspondence to a subset of integers of S whose sum is $m/2$. Eventually, note that the graph $\overline{\mathsf{GG}}(\mathcal{A}(S))$ is acyclic since it contains the edge over α and S' only. □

5. Tractable Winner-Determination

Now that the complexity of the basic Feasibility problem has been analyzed, we can turn to isolate tractable classes for the Winner-Determination problem. To this end, we first observe that all the hardness results we have derived for Feasibility as well as all hardness results for classical combinatorial auctions (see, e.g., [23]) are inherited by Winner-Determination on MMUCAs. Thus, in order to isolate classes of instances where Winner-Determination is tractable, we must avoid the sources of complexities that emerged so far in our analysis or that are known from the literature.

5.1. Bounded intricacy and (acyclic) hypergraph encodings

Let $\mathcal{A}_C = \langle G, \{\mathfrak{L}_1, \dots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ be a MMUCA instance, let $\mathfrak{B} = \bigcup_{\ell=1}^{n} bids(\mathfrak{L}_\ell)$ be the multi-set of all its atomic bids, and let $\mathcal{T}_r$ be the multi-set of all transformations occurring in $\mathfrak{B}$. For the sake of presentation, each element in $\mathfrak{B}$ (resp., $\mathcal{T}_r$) is associated with a unique identifier $\mathbf{b}_i$ (resp., t_j) with $1 \leqslant i \leqslant |\mathfrak{B}|$ (resp., $1 \leqslant j \leqslant |\mathcal{T}_r|$), so that we can conveniently refer to sets of identifiers rather than to multi-sets of elements. Then, the *auction hypergraph* $\mathsf{AH}(\mathcal{A}_C) = (N, H)$ is the hypergraph whose set N of nodes is such that $N = T \cup B$, where $T = \{t_1, \dots, t_{|\mathcal{T}_r|}\}$ and $B = \{\mathbf{b}_1, \dots, \mathbf{b}_{|\mathfrak{B}|}\}$, and whose hyperedges in H are defined as follows:

(1) For each good $g \in G$, H contains the hyperedge $\{t_j \mid t_j = \langle \mathcal{I}, \mathcal{O}\rangle \in \mathcal{T}_r$ and $\mathcal{I}(g) + \mathcal{O}(g) > 0\}$ over the transformations requiring or producing g.

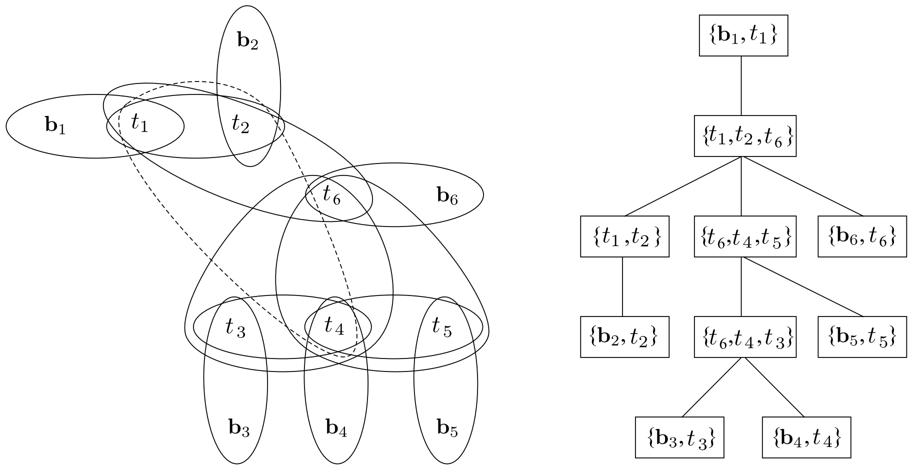

Fig. 9. Auction hypergraph in Example 5.1, and a join tree for it.

(2) For each atomic bid $\mathbf{b}_i = \langle \mathcal{B}, p, \textsc{type}\rangle \in \mathfrak{B}$ and for each transformation $t_j \in \mathcal{B}$, H contains the hyperedge $\{\mathbf{b}_i, t_j\}$.
(3) If C=XOR, then for each bid $\mathfrak{L}_\ell$, with $\ell \in \{1, \dots, n\}$, H contains the hyperedge $\{\mathbf{b}_i \mid \mathbf{b}_i \in \mathit{bids}(\mathfrak{L}_\ell)\}$ over the atomic bids involved in it.

Note that the above encoding generalizes the one proposed in [6] for classical combinatorial auctions, which in fact consists of the hyperedges of type (1) only.

Example 5.1. Consider the MMUCA instance $\bar{\mathcal{A}} = \langle \bar{G}, \bar{\mathfrak{B}}, \bar{\mathcal{U}}_{in}, \bar{\mathcal{U}}_{out}\rangle$ discussed in Example 2.2. Recall that $\{t_1, \dots, t_6\}$ is the set of all transformations in $\bar{\mathcal{A}}$, and let $\mathbf{b}_i$ denote the bid over the transformation t_i, for each $i \in \{1, \dots, 6\}$. The associated auction hypergraph AH($\bar{\mathcal{A}}$) is shown in Fig. 9—for the moment, ignore the dashed hyperedge. Note, for instance, that AH($\bar{\mathcal{A}}$) contains the hyperedge $\{t_1, t_2\}$, as these two transformations both produce as output a drive train. ◁

An important structural property of hypergraphs is acyclicity. A hypergraph $\mathcal{H}$ is *acyclic* iff it has a join tree [24]. A *join tree JT*($\mathcal{H}$) for a hypergraph $\mathcal{H}$ is a tree whose vertices are the hyperedges of $\mathcal{H}$ such that, whenever the same node $X \in V$ occurs in two hyperedges h_1 and h_2 of $\mathcal{H}$, then X occurs in each vertex on the unique path linking h_1 and h_2 in $JT(\mathcal{H})$. Note that the notion of acyclicity we use here is the most general one in the literature, also known as α-acyclicity. As an example, the hypergraph on the left of Fig. 9 is acyclic, as it is witnessed by the join tree on the right.

Unfortunately, the acyclicity of the associated auction hypergraph is not *alone* a guarantee for tractability. For instance, one can easily check that the NP-hardness proof of Theorem 4.3 is established over a class of instances whose underlying auction hypergraphs are acyclic. As the proof is based on scenarios where a “large” number of transformations produce or consume the same “small” set of goods, it is natural to look for combining acyclicity with a further bound on the number of transformations producing a given good g (as to control its availability in any solution), plus either a bound on the maximum quantity of g that can be produced over all solutions, or a bound on the number of transformations requiring g (as to control the consumption of g).

The above bounds are formalized via the measure of *intricacy* of an instance $\mathcal{A}_C = \langle G, \{\mathfrak{L}_1, \dots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$, denoted by $\mathit{intr}(\mathcal{A}_C)$, which we define as the value:

$$\max_{g \in G}\left(\left|\left\{\langle \mathcal{I}, \mathcal{O}, p\rangle \mid \mathcal{O}(g) > 0\right\}\right| + \min\left\{\mathcal{U}_{in}(g) + \sum_{\langle \mathcal{I}, \mathcal{O}, p\rangle \in \mathcal{T}_r} \mathcal{O}(g), \left|\left\{\langle \mathcal{I}, \mathcal{O}, p\rangle \mid \mathcal{I}(g) > 0\right\}\right|\right\}\right).$$

The main result in this section is to show that MMUCA instances with (nearly)acyclic auction hypergraphs and with intricacy bounded by some fixed natural number are tractable. The result is shown by encoding MMUCA instances in terms of *constraint satisfaction problem* (CSP) instances, and by subsequently exploiting known structural tractability results in this latter setting. Note that the technique is essentially the one adopted by [6] to solve classical combinatorial auctions over structurally-restricted classes of instances (cf. [25,26]), and it is here extended to deal with MMUCAs.

5.2. From MMUCAs to CSP instances

We start by recalling some preliminaries on constraint satisfaction. The reader interesting in expanding on this formalism is referred to [27].

A CSP instance is a triple $\mathcal{J} = \langle \mathit{Var}, U, \mathbf{C}\rangle$, where Var is a finite set of variables, U is a finite domain of values, and $\mathbf{C} = \{C_1, C_2, \dots, C_q\}$ is a finite set of constraints. Each constraint C_v, for $1 \leqslant v \leqslant q$, is a pair (S_v, r_v), where $S_v \subseteq \mathit{Var}$ is a set of variables called the *constraint scope*, and r_v is a set of substitutions from variables in S_v to values in U indicating the allowed combinations of simultaneous values for the variables in S_v. A substitution from a set of variables $V \subseteq Var$ to U is extensively denoted as the set of pairs of the form X/u, where $u \in U$ is the value to which $X \in V$ is mapped. The restriction of a substitution $\theta : V \to U$ over a set $S \subseteq V$ of variables is denoted by $\theta[S]$. A solution to $\mathcal{J}$ is a substitution $\theta : Var \to U$ such that $\theta[S_v] \in r_v$ holds, for each v with $1 \leqslant v \leqslant q$. The structure of a CSP instance $\mathcal{J}$ is best represented by its associated hypergraph $\mathcal{H}(\mathcal{J}) = (V, H)$, where $V = Var$ and $H = \{S \mid (S, r) \in \mathcal{C}\}$.

Given the instance $\mathcal{A}_C = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ where $\mathfrak{B}$ is the multi-set of all atomic bids and $\mathcal{T}_r$ is the multi-set of all transformations occurring in $\mathfrak{B}$, we define $CSP(\mathcal{A}_C) = \langle Var, U, \mathbf{C}\rangle$ as the CSP instance such that:

- Variables are associated with (the identifiers of) the transformations and the bids in $\mathcal{A}_C$, that is, $Var = \{t_1, \ldots, t_{|\mathcal{T}_r|}\} \cup \{\mathbf{b}_1, \ldots, \mathbf{b}_{|\mathfrak{B}|}\}$.
- The domain U consists of the set $\{0, 1, \ldots, |\mathcal{T}_r|\}$ of natural numbers. Accordingly, substitutions from Var to U have the following intuitive meaning. A "transformation" variable t_j taking value $\alpha > 0$ (resp., $\alpha = 0$) means that t_j is executed at the α-th step of a sequence witnessing the existence of a solution to $\mathcal{A}_C$ (resp., is not executed). A "bid" variable $\mathbf{b}_i$ taking value 1 (resp., 0) means that $\mathbf{b}_i$ is accepted (resp., not accepted) by the auctioneer. In the following, for a substitution θ, let σ_θ be the sequence of transformations such that t_j occurs at the α-th position of σ_θ (resp., does not occur in it) if $t_j/\alpha \in \theta$ and $\alpha > 0$ (resp., $\alpha = 0$). Moreover, let $\mathfrak{B}_\theta$ be the set of bids $\{\mathbf{b}_i \mid \mathbf{b}_i/1 \in \theta\}$.
- The set $\mathcal{C}$ contains constraints of four types:
  (1) For each good $g \in G$, $\mathcal{C}$ contains the constraint $C_g = (S_g, r_g)$ such that $S_g = \{t_j \mid t_j = \langle \mathcal{I}, \mathcal{O}\rangle \in \mathcal{T}_r$ and $\mathcal{I}(g) + \mathcal{O}(g) > 0\}$ and where r_g is defined as follows. For a substitution θ, we say that $\sigma_\theta = \langle \mathcal{I}_1, \mathcal{O}_1\rangle, \ldots, \langle \mathcal{I}_r, \mathcal{O}_r\rangle$ is a *feasible outcome for $\mathcal{A}_C$ w.r.t. g* if $m_{c-1} \geqslant \mathcal{I}_c(g)$, for each $c \in \{1, \ldots, r\}$, where $m_0 = \mathcal{U}_{in}(g)$ and $m_c = m_{c-1} + \mathcal{O}_c(g) - \mathcal{I}_c(g)$. In the case of lack of free disposal (resp., under the free disposal assumption), we furthermore require that $m_r = \mathcal{U}_{out}(g)$ (resp., $m_r \geqslant \mathcal{U}_{out}(g)$) holds. Then, r_g consists of all substitutions $\theta : S_g \to U$ where σ_θ is a feasible outcome for $\mathcal{A}_C$ w.r.t. g. Note that if the intricacy $intr(\mathcal{A}_C)$ is bounded by some given fixed constant, then the number of substitutions in r_g, i.e., the size $|r_g|$, is polynomial.
  (2_a) For each atomic bid $\mathbf{b}_i = \langle \mathcal{B}, p, \text{PARTIAL}\rangle \in \mathfrak{B}$ and each transformation $t_j \in \mathcal{B}$, $\mathcal{C}$ contains the constraint $C_{\mathbf{b}_i,t_j} = (S_{\mathbf{b}_i,t_j}, r_{\mathbf{b}_i,t_j})$ such that $S_{\mathbf{b}_i,t_j} = \{\mathbf{b}_i, t_j\}$ and where $r_{\mathbf{b}_i,t_j} = \{\{\mathbf{b}_i/0, t_j/0\}, \{\mathbf{b}_i/1, t_j/0\}, \{\mathbf{b}_i/1, t_j/1\}, \ldots, \{\mathbf{b}_i/1, t_j/|\mathcal{T}_r|\}\}$. Note that $|r_{\mathbf{b}_i,t_j}| = 2 + |\mathcal{T}_r|$.
  (2_b) For each atomic bid $\mathbf{b}_i = \langle \mathcal{B}, p, \text{FULL}\rangle \in \mathfrak{B}$ and each transformation $t_j \in \mathcal{B}$, $\mathcal{C}$ contains the constraint $C_{\mathbf{b}_i,t_j} = (S_{\mathbf{b}_i,t_j}, r_{\mathbf{b}_i,t_j})$ such that $S_{\mathbf{b}_i,t_j} = \{\mathbf{b}_i, t_j\}$ and where $r_{\mathbf{b}_i,t_j} = \{\{\mathbf{b}_i/0, t_j/0\}, \{\mathbf{b}_i/1, t_j/1\}, \ldots, \{\mathbf{b}_i/1, t_j/|\mathcal{T}_r|\}5\}$. Note that $|r_{\mathbf{b}_i,t_j}| = 1 + |\mathcal{T}_r|$.
  (3) If C = XOR, then for each bid $\mathfrak{L}_\ell$, with $\ell \in \{1, \ldots, n\}$, $\mathcal{C}$ contains the constraint $C_{\mathfrak{L}_\ell} = (S_{\mathfrak{L}_\ell}, r_{\mathfrak{L}_\ell})$ such that $S_{\mathfrak{L}_\ell} = \{\mathbf{b}_i \mid \mathbf{b}_i \in bids(\mathfrak{L}_\ell)\}$ and where $r_{\mathfrak{L}_\ell}$ contains the substitution where each variable is mapped to 0, plus all possible substitutions where exactly one variable is mapped to 1 and the remaining ones are mapped to 0. Note that $|r_{\mathfrak{L}_\ell}| = 1 + n$.

The crucial properties of the above correspondence from MMUCA instances to CSP instances are formalized below.

Lemma 5.2. *Let $\mathcal{A}_C$ be a MMUCA instance. Then, the following properties hold*:

(A) *If $intr(\mathcal{A}_C) \leqslant k$, for some fixed natural number $k > 0$, then $CSP(\mathcal{A}_C)$ can be built in polynomial time; in particular, each constraint relation contains polynomially many substitutions.*
(B) $\mathsf{AH}(\mathcal{A}_C) = \mathcal{H}(CSP(\mathcal{A}_C))$*, i.e., the hypergraph associated with $\mathcal{A}_C$ coincides with the hypergraph associated with $CSP(\mathcal{A}_C)$.*
(C) *If $\mathfrak{B}'$ is a solution to $\mathcal{A}_C$, then there is a solution θ to $CSP(\mathcal{A}_C)$ such that $\mathfrak{B}_\theta = \mathfrak{B}'$.*
(D) *If θ is a solution to $CSP(\mathcal{A}_C)$, then $\mathfrak{B}_\theta$ is a solution to $\mathcal{A}_C$.*

Proof. Property (A) can be easily seen to hold by checking the sizes of the various constraint relations, as we have already pointed out while defining constraints of types (1), (2_a), (2_b), and (3). The fact that $\mathsf{AH}(\mathcal{A}_C) = \mathcal{H}(CSP(\mathcal{A}_C))$ holds straightforwardly follows from the definition of the constraints scopes of $CSP(\mathcal{A}_C)$, which precisely coincides with the hyperedges of $\mathsf{AH}(\mathcal{A}_C)$—scopes of type ($2_a$) and ($2_b$) correspond to hyperedges of type (2), whereas scopes of types (1) and (3) correspond to hyperedges of types (1) and (3), respectively. Then, we have to focus on showing the correctness of the encoding, established via (C) and (D). Let us first consider the case where $\mathcal{A}_C = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out}\rangle$ is such that $\mathfrak{L}_\ell$ is an atomic bid, for each $\ell \in \{1, \ldots, n\}$.

(C) Assume that $\mathfrak{B}'$ is a solution to $\mathcal{A}_C$, and let $\sigma = \langle \mathcal{I}_1, \mathcal{O}_1\rangle, \ldots, \langle \mathcal{I}_r, \mathcal{O}_r\rangle$ be the sequence of transformations associated with $\mathfrak{B}'$ and satisfying the conditions illustrated in Definition 2.1. Consider the substitution $\theta : Var \to U$ for $CSP(\mathcal{A}_C) = \langle Var, U, \mathbf{C}\rangle$ such that $\sigma_\theta = \sigma$, $\mathfrak{B}_\theta = \mathfrak{B}'$, and $\mathbf{b}_i/0 \in \theta$, for each $\mathbf{b}_i \notin \mathfrak{B}'$. We claim that θ is a solution to $CSP(\mathcal{A}_C)$, i.e., it satisfies constraints of types (1), (2_a), and (2_b). Indeed, because of condition (3) in Definition 2.1, σ_θ is a feasible outcome for $\mathcal{A}_C$ w.r.t. any good g, and hence $\theta[S_g] \in r_g$. That is, constraints of type (1) are satisfied. Concerning the constraints of type (2_a), just notice that, by condition (2) in Definition 2.1, for each transformation $t_j/\alpha \in \theta$ with $\alpha > 0$ belonging to a bid $\mathbf{b}_i = \langle \mathcal{B}, p, \text{PARTIAL}\rangle$, we have that $\mathbf{b}_i/1 \in \theta$. Thus, these constraints are also satisfied. Finally, note that by condition (1) in Definition 2.1, for each atomic bid $\mathbf{b}_i = \langle \mathcal{B}, p, \text{FULL} \rangle \in \mathfrak{B}$, we have that $\mathbf{b}_i \in \mathcal{B}'$ can hold if, and only if, t_j occurs in σ for each transformation $t_j \in \mathcal{B}$. Thus, $\theta[\{\mathbf{b}_i\}] = 1$ (resp., $\theta[\{\mathbf{b}_i\}] = 0$) implies that $\theta[\{t_j\}] > 0$ (resp., $\theta[\{t_j\}] = 0$). It follows that $\theta[S_{\mathbf{b}_i, r_j}] \in r_{\mathbf{b}_i, t_j}$. Hence, constraints of type (2_b) are also satisfied.

(D) Let θ be a solution to *CSP*($\mathcal{A}_C$). Because of the constraints of types (1), (2_a), and (2_b) we have that $\mathfrak{B}_\theta$ (and the associated sequence σ_θ) satisfies conditions (3), (2), and (1) in Definition 2.1, respectively. Indeed, constraints of type (1) enforce that σ_θ is a feasible outcome for $\mathcal{A}_C$ w.r.t. any good g. Constraints of type (2_a) guarantee that if a transformation t_j occurs in σ_θ and is such that $t_j \in \mathcal{B}$ with $\mathbf{b}_i = \langle \mathcal{B}, p, \text{TYPE} \rangle$, then the bid $\mathbf{b}_i$ belongs to $\mathfrak{B}_\theta$. And, finally, constraints of type (2_b) enforce that if a bid $\mathbf{b}_i = \langle \mathcal{B}, p, \text{FULL} \rangle$ occurs in $\mathfrak{B}_\theta$, then all its transformations belong to the sequence σ_θ. Thus, $\mathfrak{B}_\theta$ is a solution to $\mathcal{A}_C$.

To complete the proof, we have to consider the cases where $\mathcal{A}_C$ is defined over OR-bids and XOR-bids. In fact, the former case precisely coincides with the scenario where all bids are atomic, as OR-bids do not actually provide further constraints. For the latter case, instead, we have just to notice that constraints of type (3) precisely guarantee that for each bid $\mathfrak{L}_\ell$, with $\ell \in \{1, \ldots, n\}$, at most one atomic bid in $\mathfrak{L}_\ell$ can be accepted by the auctioneer. Thus, they correctly enforce the semantics of XOR-bids. □

With the above lemma in place, the final ingredient we need is to equip CSP instances with weights as to properly encode the goal of the auctioneer wishing to the maximize the revenue over all the possible solutions.

An instance of a *constraint satisfaction optimization problem* (CSOP) consists of a pair $\langle \mathcal{J}, w \rangle$, where $\mathcal{J} = \langle Var, U, \mathcal{C} \rangle$ is a CSP instance and where $w : Var \times U \to \mathbb{R}$ is a *weighting function* mapping substitutions for individual variables to real numbers, which we assume to be explicitly listed in the input. For a substitution $\{X_1/u_1, \ldots, X_n/u_n\}$, we denote by $w(\{X_1/u_1, \ldots, X_n/u_n\})$ the value $\sum_{i=1}^n w(X_i, u_i)$. Then, a solution to a CSOP instance $\langle \mathcal{J}, w \rangle$ is a solution θ to $\mathcal{J}$ such that $w(\theta) \geqslant w(\theta')$, for each solution θ' to $\mathcal{J}$. The hypergraph $\mathcal{H}(\langle \mathcal{J}, w \rangle)$ associated with $\langle \mathcal{J}, w \rangle$ is $\mathcal{H}(\mathcal{J})$.

Given the instance $\mathcal{A}_C = \langle G, \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$, we now define *CSOP*($\mathcal{A}_C$) as the constraint satisfaction optimization problem over the instance *CSP*($\mathcal{A}_C$) and the weighting function $w(\mathcal{A}_C)$ such that: $w(\mathcal{A}_C)(\mathbf{b}_i, 1) = p$, for each atomic bid $\mathbf{b}_i = \langle \mathcal{B}, p, \text{TYPE} \rangle$ in $\mathcal{A}_C$, and $w(\mathcal{A}_C)(X, u) = 0$ for each other substitution X/u. Note that the following is immediate, by Lemma 5.2 and by the construction of *CSOP*($\mathcal{A}_C$).

Lemma 5.3. *Let $\mathcal{A}_C$ be a MMUCA instance. Then, the following properties hold*:

- *If $\mathfrak{B}'$ is an optimal solution to $\mathcal{A}_C$, then there is a solution θ to CSOP($\mathcal{A}_C$) such that $\mathfrak{B}_\theta = \mathfrak{B}'$.*
- *If θ is a solution to CSOP($\mathcal{A}_C$), then $\mathfrak{B}_\theta$ is an optimal solution to $\mathcal{A}_C$.*

We can eventually conclude the exposition by showing the tractability of MMUCA instances over acyclic structures having bounded intricacy.

Theorem 5.4. *For any fixed natural number $k > 0$,* WINNER-DETERMINATION *can be solved in polynomial time on any class of MMUCA instances $\mathcal{A}_C$ such that* $\mathsf{AH}(\mathcal{A}_C)$ *is acyclic and intr($\mathcal{A}_C$) $\leqslant k$.*

Proof. After Lemma 5.2, it is immediate that if the intricacy is bounded by some fixed natural number, then we can build in polynomial time a constraint satisfaction optimization problem instance *CSOP*($\mathcal{A}_C$), which is in particular such that $\mathsf{AH}(\mathcal{A}_C)$ is acyclic if, and only if, *CSOP*($\mathcal{A}_C$) is acyclic. Solutions to acyclic CSOP instances can be computed in polynomial time (see, e.g., [6,25,26]). The result then follows as solutions to *CSOP*($\mathcal{A}_C$) one-to-one correspond with optimal solutions to $\mathcal{A}_C$, by Lemma 5.3. □

5.3. Beyond acyclic hypergraphs

Many attempts have been made in the literature for extending the good results about acyclic CSP instances to relevant classes of *nearly acyclic* structures. We call these techniques *structural decomposition methods*, because they are based on the "acyclicization" of cyclic (hyper)graphs. Indeed, these methods are aimed at transforming any given cyclic instance into an equivalent acyclic one, by organizing its atoms or variables into a polynomial number of clusters, and by arranging these clusters as a tree, called *decomposition tree*. The original instance is then evaluated via this tree, with a cost that is exponential in the cardinality of the largest cluster, also called *width* of the decomposition, and polynomial if the width is bounded by some constant.

In fact, those positive results for constraint satisfaction that hold over acyclic instances can be straightforwardly extended to classes of nearly acyclic ones for which a decomposition tree can be efficiently computed. As an important application, we consider here the *hypertree decomposition* [28].

For any hypergraph $\mathcal{H}$, we denote its nodes and edges by $\mathcal{N}(\mathcal{H})$ and $\mathcal{E}(\mathcal{H})$, respectively. A *hypertree for a hypergraph $\mathcal{H}$* is a triple $\langle T, \chi, \lambda \rangle$, where $T = (N, E)$ is a rooted tree, and χ and λ are labeling functions, which associate each vertex $p \in N$ with two sets $\chi(p) \subseteq \mathcal{N}(\mathcal{H})$ and $\lambda(p) \subseteq \mathcal{E}(\mathcal{H})$. For a set of edges $H \subseteq \mathcal{E}(\mathcal{H})$, $\mathcal{N}(H)$ denotes the set $\bigcup_{h \in H} h$. If $T' = (N', E')$

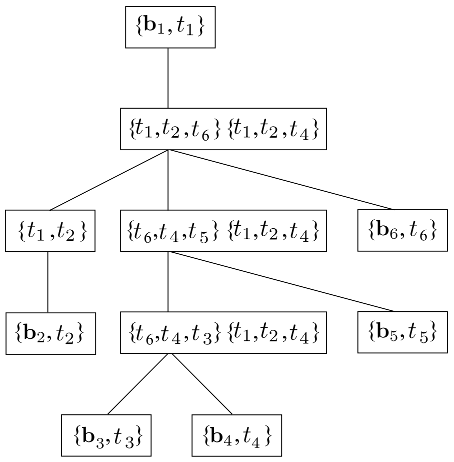

Fig. 10. A hypertree decomposition $HD = \langle T, \chi, \lambda\rangle$ of the hypergraph in Fig. 9. Each vertex p of T reports the λ-labeling only, and we set $\chi(p) = \mathcal{N}(\lambda(p))$.

is a subtree of T, we define $\chi(T') = \bigcup_{v \in N'} \chi(v)$. We denote the set of vertices N of T by *vertices*(T). Moreover, for any $p \in N$, T_p denotes the subtree of T rooted at p.

Definition 5.5. (See [28].) A *hypertree decomposition* of a hypergraph $\mathcal{H}$ is a hypertree $HD = \langle T, \chi, \lambda\rangle$ for $\mathcal{H}$ satisfying the following conditions:

(1) for each edge $h \in \mathcal{E}(\mathcal{H})$, there exists $p \in vertices(T)$ such that $h \subseteq \chi(p)$, and $h \in \lambda(p)$;
(2) for each node $Y \in \mathcal{N}(\mathcal{H})$, the set $\{p \in vertices(T) \mid Y \in \chi(p)\}$ induces a (connected) subtree of T;
(3) for each $p \in vertices(T)$, $\chi(p) \subseteq \mathcal{N}(\lambda(p))$;
(4) for each $p \in vertices(T)$, $\mathcal{N}(\lambda(p)) \cap \chi(T_p) \subseteq \chi(p)$.

The *width* of a hypertree decomposition $\langle T, \chi, \lambda\rangle$ is $\max_{p \in vertices(T)} |\lambda(p)|$. The *hypertree width* $hw(\mathcal{H})$ of $\mathcal{H}$ is the minimum width over all its hypertree decompositions. $\square$

Example 5.6. Let $\bar{\mathcal{A}}'$ be the MMUCA obtained by modifying the instance $\bar{\mathcal{A}}$ in Example 5.1 in a way that transformations t_1 and t_2 both produce as an output a dynamo, and by leaving unchanged all the other transformations and bids. As t_4 also produces a dynamo, the auction hypergraph $\mathsf{AH}(\bar{\mathcal{A}}')$ is the one shown in Fig. 9, including the dashed hyperedge modeling the fact that t_1, t_2 and t_4 interact by producing the some output. Note that the modified hypergraph is not acyclic. A 2-width hypertree decomposition $HD = \langle T, \chi, \lambda\rangle$ of $\mathsf{AH}(\bar{\mathcal{A}}')$ is reported in Fig. 10. $\triangleleft$

The classes of CSP instances having bounded hypertree width have the same desirable computational properties as acyclic CSPs [29]. Indeed, from a CSP instance $\mathcal{J} = \langle Var, U, \mathbf{C}\rangle$ and a hypertree decomposition HD of $\mathcal{H}(\mathcal{J})$ of width h, we may build an acyclic CSP instance $\mathcal{J}' = \langle Var, U, \mathbf{C}\rangle$ with the same solutions as $\mathcal{J}$. The overall cost of deciding whether $\mathcal{J}$ is satisfiable is in this case $O(m \times h \times r_{max}^h \times \log r_{max})$, where r_{max} denotes the size of the largest constraint relation and m is the number of vertices of the decomposition tree, with $m \leqslant |Var|$ (in that we may always find decompositions in a suitable normal form without redundancies, so that the number of vertices in the tree cannot exceed the number of variables of the given instance). To be complete, if the input consists of $\mathcal{J}$ only, we have to compute the decomposition, too. This can be done with a guaranteed polynomial-time upper bound in the case of hypertree decompositions [28]. Putting these arguments together with Theorem 5.4, the following result is established.

Theorem 5.7. *For any pair of fixed natural numbers $h > 0$ and $k > 0$,* Winner-Determination *can be solved in polynomial time on any class of MMUCA instances $\mathcal{A}_\mathsf{C}$ such that $hw(\mathsf{AH}(\mathcal{A}_\mathsf{C})) \leqslant h$ and $intr(\mathcal{A}_\mathsf{C}) \leqslant k$.*

6. Conclusion

The problem of identifying tractability islands for mixed multi-unit combinatorial auctions has been faced, by complementing tractability results that were obtained in the literature for classical combinatorial auctions. In particular, a clear picture of the computational complexity of MMUCAs instances has been depicted under structural and qualitative restrictions, which characterize interactions among bidders and types of bids involved in the various transformations, respectively.

The analysis of the computational complexity of MMUCAs is an important prerequisite to define efficient solution approaches. On the one hand, from our analysis, a number of settings emerged to be NP-hard. In these cases, it is unlikely that polynomial-time algorithms exist and, hence, the question comes into play about whether it is possible to define suitable heuristics that well-behave in practice. On the other hand, a significant number of other settings turned out to be tractable. In these cases, it has to be noticed that our proofs are constructive and, hence, immediately provide efficient solution algorithms, which would be relevant to implement and test on real scenarios.

Moreover, it is relevant to point out that the problems we analyzed in the context of supply-chain formation via MMUCAs can be abstractly seen as *synthesis* problems, where bids can be seen as playing the role of computational units (i.e., the tasks to be executed) and goods can be seen as representing descriptions, such as ontological ones, for their inputs and outputs. These problems frequently occur in the context of *service oriented computing*, where software applications are built by automatically composing and configuring existing (web) services (see, e.g., [30–32]). More generally, similar issues emerge in the synthesis of workflow models [33]. This is a problem that is currently receiving renewed interest because of its application to workflow management systems supporting declarative process models, where users are just in charge of defining the goal of the process in terms of a description of the required outputs as well as of defining the pre-conditions and post-conditions for the various tasks [34]. Eventually, the results discussed in this paper can be also applied to analyze biological processes [35] in general, and metabolic pathways [36] in particular, which encode biochemical reactions involving enzymes and metabolites. A metabolic reaction requires as input some metabolites and produces as output other metabolites. In this context, metabolic reactions can be mapped to bids and metabolites to goods.

Therefore, while being stated in the context of MMUCAs, the results we derived in this paper found application in the above settings, too. In particular, the techniques we have defined to identify islands of tractability based on structural restrictions can be used to identify islands of tractability in such synthesis problems, too. Moreover, concerning such structural tractability issues, we point out that our approach is based on encoding the semantics underlying MMUCAs in terms of constraint satisfaction problems. In fact, CSPs are "static" in nature, and therefore an ad-hoc encoding approach has been proposed to deal with the dynamism of our setting. Note that this approach can be well used to model via CSP instances different dynamic environments, where in particular the analysis is restricted to a polynomially-bounded or even just fixed time horizon. This is, for instance, the case of certain planning problems (see, e.g., [37]) or of reasoning problems over acyclic process models.

From a technical viewpoint, there are a few issues that remained open, namely to check whether strongly NP-completeness results can be proven in Theorem 3.10 and Theorem 4.3, and to assess whether FEASIBILITY is tractable on the class $\mathcal{C}_{\mathrm{OR}}(1, 1, k, k, 1, k)$, for some fixed constant k. From a more general perspective, instead, it would be interesting to embark on the implementation of the approaches we have exhibited to single out islands of tractability, and on the definition of solution algorithms for arbitrary MMUCA instances, which take advantage of structural and qualitative properties that hold over portions of them. In addition, an other interesting avenue of further research is to analyze the complexity of MMUCAs for more involved kinds of bidding languages, where for instance bidders' preferences or time constraints [19] are taken into account.

Appendix A

Lemma 3.2. SATISFIABILITY *is* NP*-hard, even on classes of Boolean formulas in conjunctive normal form where each variable occurs positively in two clauses and negatively in another, and where each clause contains three literals at most.*

Proof. Recall that SATISFIABILITY is NP-hard, even if each clause contains three literals at most [21]. Moreover, w.l.o.g., we can assume that the same variable occurs at most once in each clause, and that each variable occurs at least in two clauses. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula in conjunctive normal over the variables $X_1, \ldots, X_n$ form satisfying the above conditions. Based on Φ, we have to show how to build an equivalent formula Φ' (i.e., such that Φ is satisfiable if, and only if, Φ' is satisfiable) where each variable occurs positively in two clauses and negatively in another, and where each clause still contains three literals at most.

For each $1 \leqslant j \leqslant m$, let c_j be a clause of Φ and let c'_j be the clause obtained by replacing in it each variable X_i with the fresh variable X_i^k, where k is the number of clauses belonging to the set $\{c_1, \ldots, c_j\}$ where X_i occurs (either positively or negatively). Moreover, for each variable X_i, let $m_i \geqslant 2$ be the total number of clauses where it occurs and, for each $1 \leqslant i \leqslant n$ and $1 \leqslant k \leqslant m_i$, let $c_{i,k}$ be the clause $(X_i^k \vee \neg X_i^{(k+1) \bmod m_i})$. Finally, consider the formula Φ' such that

$$\Phi' = c'_1 \wedge \cdots \wedge c'_m \wedge \bigwedge_{i=1}^{n} \bigwedge_{k=1}^{m_i} c_{i,k}.$$

By construction, Φ' is now defined over the variables in the set $\{X_i^k \mid 1 \leqslant i \leqslant n, 1 \leqslant k \leqslant m_i\}$. Moreover, in any satisfying truth assignment either all variables in $\{X_i^1, \ldots, X_i^{m_i}\}$, for each $1 \leqslant i \leqslant m$, evaluate true or they all evaluate false, because of the clauses of the form $c_{i,k}$. Thus, Φ and Φ' are equivalent.

Eventually, to conclude the proof, observe that each variable X_i^k occurs in exactly three clauses. In particular, it occurs positively in $c_{i,k}$ and negatively in $c_{i,(m_i+k-1) \bmod m_i}$. Moreover, X_i^k occurs in a clause of the form c'_j, either positively or negatively. It follows that if there is no variable of the form X_i^k that negatively occurs in a clause of the form c'_j, then Φ'

is already such that each variable occurs positively in two clauses and negatively in another. Otherwise, for each variable X_i^k occurring negatively in a clause of the form c'_j, we have just to substitute each occurrence of X_i^k (resp., $\neg X_i^k$) with $\neg X_i^k$ (resp., X_i^k). The resulting formula is, of course, still equivalent to Φ, while satisfying the condition that each variable occurs positively in two clauses and negatively in another. As a final remark, note that all clauses contain three literals at most. □

Lemma 3.7. EXACT-1 SATISFIABILITY *is* NP-*hard, even on classes of Boolean formulas in conjunctive normal form where each variable occurs positively in two clauses and negatively in another, and where each clause contains three literals at most.*

Proof. Recall that EXACT-1 SATISFIABILITY is NP-hard, even if each clause exactly contains three literals (problem ONE-IN-THREE 3SAT in [21]), and hence is each clause contains three literals at most. W.l.o.g., assume that the same variable occurs at most once in each clause, and that each variable occurs at least in two clauses. Let $\Phi = c_1 \wedge \cdots \wedge c_m$ be a Boolean formula in conjunctive normal over the variables $X_1, \ldots, X_n$ form satisfying the above conditions. Let Φ' be the Boolean formula built in the proof of Lemma 3.2 over the variables in $\{X_i^k \mid 1 \leqslant i \leqslant n, 1 \leqslant k \leqslant m_i\}$, and recall that Φ' and Φ are equivalent, that each variable occurs in Φ' positively in two clauses and negatively in another, and that each clause in Φ' contains three literals at most.

Observe now that in any satisfying assignment for Φ', each clause of the form $c_{i,k}$, for each $1 \leqslant i \leqslant n$ and $1 \leqslant k \leqslant m_i$, contains one literal that evaluates true and another that evaluate false. Recall also that in any satisfying truth assignment, either all variables in $\{X_i^1, \ldots, X_i^{m_i}\}$, for each $1 \leqslant i \leqslant m$, evaluate true or they all evaluate false. By combining the above two observations, we can conclude that there is a satisfying assignment for Φ' such that, for each clause, exactly one literal evaluates true if, and only if, there is a satisfying assignment to Φ enjoying the same property. □

References

[1] J. Cerquides, U. Endriss, A. Giovannucci, J.A. Rodríguez-Aguilar, Bidding languages and winner determination for mixed multi-unit combinatorial auctions, in: Proc. of the 20th International Joint Conference on Artificial Intelligence (IJCAI), 2007, pp. 1221–1226.
[2] W. Walsh, M. Wellman, Decentralized supply chain formation: a market protocol and competitive equilibrium analysis, Journal of Artificial Intelligence Research 19 (2003) 513–567.
[3] A. Giovannucci, M. Vinyals, J.A. Rodríguez-Aguilar, J. Cerquides, Computationally-efficient winner determination for mixed multi-unit combinatorial auctions, in: Proc. of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2008, pp. 1071–1078.
[4] B. Ottens, U. Endriss, Comparing winner determination algorithms for mixed multi-unit combinatorial auctions, in: Proc. of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2008, pp. 601–1604.
[5] V. Conitzer, J. Derryberry, T. Sandholm, Combinatorial auctions with structured item graphs, in: Proc. of the 19th National Conference on Artificial Intelligence (AAAI), 2004, pp. 212–218.
[6] G. Gottlob, G. Greco, On the complexity of combinatorial auctions: structured item graphs and hypertree decomposition, in: Proc. of the 8th ACM Conference on Electronic Commerce (EC), 2007, pp. 152–161.
[7] V. Fionda, G. Greco, Charting the tractability frontier of mixed multi-unit combinatorial auctions, in: Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI), 2009, pp. 134–139.
[8] N. Nisan, Bidding languages for combinatorial auctions, in: P. Cramton, Y. Shoham, R. Steinberg (Eds.), Combinatorial Auctions, MIT Press, 2006.
[9] M.H. Rothkopf, A. Pekec, R.M. Harstad, Computationally manageable combinational auctions, Management Science 44 (8) (1998) 1113–1114.
[10] T. Sandholm, Algorithm for optimal winner determination in combinatorial auctions, Artificial Intelligence 135 (2002) 1–54.
[11] M. Vinyals, A. Giovannucci, J. Cerquides, P. Meseguer, J.A. Rodríguez-Aguilar, A test suite for the evaluation of mixed multi-unit combinatorial auctions, Journal of Algorithms 63 (1–3) (2008) 130–150.
[12] P. Almajano, J. Cerquides, J.A. Rodríguez-Aguilar, Empirical hardness for mixed auctions, in: Proc. of the 13th Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA), 2009, pp. 161–170.
[13] A. Giovannucci, J.A. Rodríguez-Aguilar, J. Cerquides, U. Endriss, Winner determination for mixed multi-unit combinatorial auctions via Petri nets, in: Proc. of the 6th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2007, pp. 710–717.
[14] A. Giovannucci, J. Cerquides, U. Endriss, J.A. Rodríguez-Aguilar, A graphical formalism for mixed multi-unit combinatorial auctions, Journal of Autonomous Agents and Multi-Agent Systems 20 (3) (2010) 342–368.
[15] M. Vinyals, J. Cerquides, On the empirical evaluation of mixed multi-unit combinatorial auctions, in: Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis, AAMAS 2007 Workshop (AMEC/TADA), 2007, pp. 135–150.
[16] A. Giovannucci, J. Cerquides, U. Endriss, M. Vinyals, J.A. Rodríguez-Aguilar, B. Rosell, A mixed multi-unit combinatorial auctions test suite, in: Proc. of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2009, pp. 1389–1390.
[17] T. Sandholm, S. Suri, A. Gilpin, D. Levine, Winner determination in combinatorial auction generalizations, in: Proc. of the 1st International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2002, pp. 69–76.
[18] A. Kothari, T. Sandholm, S. Suri, Solving combinatorial exchanges: optimality via a few partial bids, in: Proc. of the 4th ACM Conference on Electronic Commerce (EC), 2003, pp. 236–237.
[19] A. Witzel, U. Endriss, Time constraints in mixed multi-unit combinatorial auctions, in: Proc. of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2010, pp. 1487–1488.
[20] C.H. Papadimitriou, Computational Complexity, Addison–Wesley, 1994.
[21] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and co., New York, 1979.
[22] J. Kleinberg, E. Tardos, Algorithm Design, Addison–Wesley, 2005.
[23] D. Lehmann, R. Müller, T. Sandholm, The winner determination problem, in: P. Cramton, Y. Shoham, R. Steinberg (Eds.), Combinatorial Auctions, MIT Press, 2006.
[24] P. Bernstein, N. Goodman, The power of natural semijoins, SIAM Journal on Computing 10 (4) (1981) 751–771.
[25] G. Greco, F. Scarcello, Structural tractability of constraint optimization, in: Proc. of 17th International Conference on Principles and Practice of Constraint Programming (CP), 2011, pp. 340–355.
[26] G. Gottlob, G. Greco, F. Scarcello, Tractable optimization problems through hypergraph-based structural restrictions, in: Proc. of 36th International Colloquium on Automata, Languages and Programming (ICALP), 2009, pp. 16–30.
[27] R. Dechter, Constraint Processing, Morgan Kaufmann, 2003.

[28] G. Gottlob, N. Leone, S. Scarcello, Hypertree decompositions and tractable queries, Journal of Computer and System Sciences 63 (3) (2002) 579–627.
[29] G. Gottlob, N. Leone, S. Scarcello, The complexity of acyclic conjunctive queries, Journal of the ACM 48 (3) (2001) 431–498.
[30] A. Marconi, M. Pistore, Synthesis and composition of Web services, in: M. Bernardo, L. Padovani, G. Zavattaro (Eds.), Formal Methods for Web Services, Springer-Verlag, 2009.
[31] S. Dustdar, W. Schreiner, A survey on web services composition, International Journal of Web and Grid Services 1 (1) (2005) 1–30.
[32] S.-C. Oh, D. Lee, S.R.T. Kumara, A comparative illustration of AI planning-based web services composition, SIGecom Exchanges 5 (5) (2006) 1551–9031.
[33] F. Friedler, K. Tarjan, Y.W. Huang, L.T. Fan, Graph-theoretic approach to process synthesis: polynomial algorithm for maximal structure generation, Computers and Chemical Engineering 17 (9) (1998) 929–942.
[34] M. Pesic, H. Schonenberg, W.M.P. van der Aalst, DECLARE: full support for loosely-structured processes, in: Proc. of the 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC), 2007, pp. 287–300.
[35] M. Peleg, I. Yeh, R. Altman, Modelling biological processes using workflow and Petri Net models, Bioinformatics 18 (6) (2002) 825–837.
[36] M. Kanehisa, S. Goto, KEGG: Kyoto encyclopedia of genes and genomes, Nucleic Acids Research 28 (1) (2000) 27–30.
[37] T. Eiter, W. Faber, N. Leone, G. Pfeifer, A. Polleres, A logic programming approach to knowledge-state planning, II: the DLV^k system, Artificial Intelligence 144 (1–2) (2003) 157–211.