Tool Auctions

Janosch Döcker
University of Tübingen
Germany

Britta Dorn
University of Tübingen
Germany

Ulle Endriss
ILLC, University of Amsterdam
The Netherlands

Ronald de Haan
ILLC, University of Amsterdam
The Netherlands

Sebastian Schneckenburger
University of Tübingen
Germany

Abstract

We introduce *tool auctions*, a novel market mechanism for constructing a cost-efficient assembly line for producing a desired set of products from a given set of goods and tools. Such tools can be used to transform one type of good into a different one. We then study the computational complexity of tool auctions in detail, using methods from both classical and parameterized complexity theory. While solving such auctions is intractable in general, just as for the related frameworks of combinatorial and mixed auctions, we are able to identify several special cases of practical interest where designing efficient algorithms is possible.

1 Introduction

"It looked to Sacharissa that the only tools a dwarf needed were his axe and some means of making fire. That'd eventually get him a forge, and with that he could make simple tools, and with those he could make complex tools, and with complex tools a dwarf could more or less make anything."

—Terry Pratchett, *The tRuth*

Auctions, and combinatorial auctions in particular, are a powerful family of market mechanisms for allocating resources, which have been studied in depth in Economics, Operations Research, and AI (Cramton, Shoham, and Steinberg 2006). We propose a novel type of combinatorial auction that allows the auctioneer to purchase a number of *tools* as well as other *goods*. She can then use these tools to produce further such tools and goods. Her objective is to pick a set of tools and then to execute a suitable sequence of actions, each consisting of an application of one of the tools she has access to, so as to eventually obtain the set of goods she desires. A secondary objective is to find a sequence that is as short as possible.

Our model may be interpreted as a restricted—and thus computationally less demanding—instance of the model of *mixed multi-unit combinatorial auctions* of Cerquides et al. (2007), which has applications in the domain of industrial supply-chain management. In a mixed auction, the auctioneer can purchase *transformations*, from *input goods* to *output goods*, to compose a sequence of such transformations to obtain the goods she desires. Our *tool auctions* are mixed auctions in which, for each transformation, there is only a single good that occurs both in the input set and the output set: the tool. For example, in the context of the transformation mapping $\{mixer, melon\}$ to $\{mixer, juice\}$, the *mixer* is the tool, while *melon* and *juice* are ordinary goods. In addition to limiting each transformation to just one tool, to control the complexity of the problem faced by the auctioneer, we also put certain restrictions on the number of non-tool goods involved in a single transformation.

Fully expressive mixed auctions are highly complex. The *winner determination problem*, i.e., the problem of finding a sequence that produces the desired set of goods at a given price, is NP-complete (Cerquides et al. 2007). Fionda and Greco (2013) have analysed this problem, as well as the related *feasibility problem*, where we ignore prices and simply want to find *some* sequence that does the job, in great detail. They were able to identify a number of so-called *tractability islands*, i.e., restrictions of the general setting where solving an auction is possible in polynomial time. For example, they were able to show that, when every transformation only admits a single input good and a single output good, then the feasibility problem is polynomial. But even permitting two output goods (and still just one input good) already turns the problem NP-complete. In general, existing tractability results only apply under very severe restrictions. The fundamental intuition inspiring our new auction model is that introducing constraints on the types of goods—tools *vs.* ordinary goods—whilst at the same time relaxing constraints on the number of goods involved in a transformation constitutes a promising trade-off between tractability and expressivity.

The core of this paper is devoted to testing this intuition. We provide a detailed complexity analysis of both the feasibility and the winner determination problem for tool auctions. While solving tool auctions, just like most other kinds of combinatorial auctions, is intractable in general, we are able to identify several special cases of practical interest where designing efficient algorithms is possible. In particular, we show that several relevant decision problems are either polynomial-time solvable or fixed-parameter-tractable.

The remainder of the paper is organised as follows. Section 2 defines the model of tool auctions we propose, explains its relationship to the existing model of mixed auctions, and elucidates on some of the design choices made. Section 3 presents our complexity results, while Section 4 discusses related work. Finally, Section 5 concludes with a brief dis-

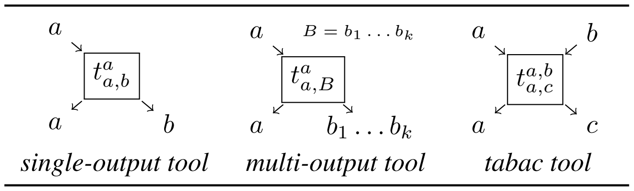

Table 1: Tools

cussion of a number of variants of our model and an outlook on possible directions for future work.

2 The Model

In this section we introduce our model of tool auctions.

2.1 Notation and Terminology

Fix a finite set G of *goods*. A *bundle* is a multiset of goods, i.e., a function $B : G \to \mathbb{N}$, which we also write as $B \in \mathbb{N}^G$, mapping each good to the multiplicity with which it occurs in the bundle (we use $\mathbb{N}$ to denote the natural numbers, together with 0). We use standard set-theoretic notation also for multisets: $x \in B$ is short for $B(x) > 0$; $B \subseteq B'$ for $\forall x \in G : B(x) \leqslant B'(x)$; $B \cup B'$ for $x \mapsto \max\{B(x), B'(x)\}$ (i.e., for the function mapping any $x \in G$ to the maximum of $B(x)$ and $B'(x)$); $B \cap B'$ for $x \mapsto \min\{B(x), B'(x)\}$; $B \setminus B'$ for $x \mapsto \max\{0, B(x) - B'(x)\}$; $|B| = k$ for $\sum_{x \in G} B(x) = k$; and $B = \emptyset$ for $|B| = 0$. We also use $\uplus$ to denote multiset addition, i.e., $B \uplus B'$ is short for $x \mapsto [B(x) + B'(x)]$. B is called a *single-unit* bundle if $|B(x)| \leqslant 1$ for all $x \in G$.

We use triples $t = \langle \tau, \mathcal{I}, \mathcal{O} \rangle \in G \times \mathbb{N}^G \times \mathbb{N}^G$ with $\tau \notin \mathcal{I} \cup \mathcal{O}$ and $\mathcal{I} \cap \mathcal{O} = \emptyset$ to describe available *transformations*, with τ being the *tool* required to use the transformation, $\mathcal{I}$ the *input goods*, and $\mathcal{O}$ the *output goods*. By a slight abuse of terminology, we use the term 'tool' not only for τ but also for t. In this paper, we focus on three types of transformations that impose certain constraints on $\mathcal{I}$ and $\mathcal{O}$: *single-output tools* with $\mathcal{I} = \emptyset$ and $|\mathcal{O}| = 1$; *multi-output tools* with $\mathcal{I} = \emptyset$; and *tabac tools* with $|\mathcal{I}| = |\mathcal{O}| = 1$. These are visualised in Table 1. In line with this visualisation, we sometimes use $t^{\tau,\mathcal{I}}_{\tau,\mathcal{O}}$ as an alternative representation for $\langle \tau, \mathcal{I}, \mathcal{O} \rangle$.

An *auction instance* is a triple $\langle \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$, where $\mathcal{T}$ is the set of transformations available to the auctioneer to choose from, $\mathcal{U}_{in} \in \mathbb{N}^G$ is the bundle she owns initially, and $\mathcal{U}_{out} \in \mathbb{N}^G$ is the bundle she hopes to end up with.

To solve an auction instance, we now consider *sequences* $\Sigma = t^1, t^2, \dots, t^\ell$ of applications of the available transformations in $\mathcal{T}$. We refer to each such t^i, which is of the form $\langle \tau^i, \mathcal{I}^i, \mathcal{O}^i \rangle$, as an *action* in the sequence. Importantly, two different actions in Σ could be applications of the same transformation (i.e., transformations can be reused). The *length* of Σ is $|\Sigma| = \ell$. We use multisets $\mathcal{M}^i \in \mathbb{N}^G$ to keep track of the goods held by the auctioneer after the ith action in Σ:

$$\begin{aligned} \mathcal{M}^0 &:= \mathcal{U}_{in}\ , \\ \mathcal{M}^i &:= (\mathcal{M}^{i-1} \setminus \mathcal{I}^i) \uplus \mathcal{O}^i \quad \text{for } i \in \{1, \dots, \ell\}. \end{aligned}$$

We call a sequence Σ *legal*, if at any stage of the process the auctioneer does in fact hold all the goods required to perform the next action, including the relevant tool, i.e., if

$$\mathcal{I}^i \subseteq \mathcal{M}^{i-1} \text{ and } \tau^i \in \mathcal{M}^{i-1} \text{ for all } i \in \{1, \dots, \ell\}.$$

We say that Σ *achieves* the desired output $\mathcal{U}_{out}$ if $\mathcal{U}_{out} \subseteq \mathcal{M}^\ell$. In this case, we write $\Sigma \rightsquigarrow \mathcal{U}_{out}$.

We are now ready to define the two decision problems we are going to analyse (see Table 2). FEASIBILITY asks whether there is a way to achieve the desired outcome. WINNER DETERMINATION (or WINDET for short) asks whether the same is possible using a sequence of at most K actions, for some $K \in \mathbb{N}$. We may think of K as a bound on time available to the auctioneer (if each action takes up one time unit) or as a budget (if each action costs one currency unit).

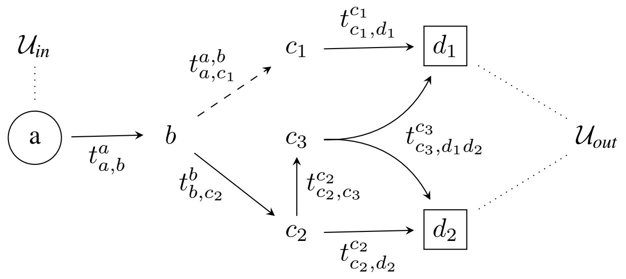

Figure 1: Visualisation of Example 1

Example 1. *Consider the goods*

$$G = \{a, b, c_1, c_2, c_3, d_1, d_2\}$$

and the auction instance $\langle \mathcal{T}, \{a\}, \{d_1, d_2\} \rangle$, *where*

$$\mathcal{T} = \{t^a_{a,b}, t^{a,b}_{a,c_1}, t^{c_1}_{c_1,d_1}, t^b_{b,c_2}, t^{c_2}_{c_2,d_2}, t^{c_2}_{c_2,c_3}, t^{c_3}_{c_3,d_1d_2}\}.$$

So the auctioneer initially only holds the good a *and wants to end up with the bundle* $\{d_1, d_2\}$. *Figure 1 visualises this scenario.* $\Sigma_1 = t^a_{a,b}, t^{a,b}_{a,c_1}, t^{c_1}_{c_1,d_1}$ *is the shortest sequence to achieve* $\{d_1\}$, *and* $\Sigma_2 = t^a_{a,b}, t^b_{b,c_2}, t^{c_2}_{c_2,d_2}$ *is the shortest one to achieve* $\{d_2\}$. *By combining the two, we obtain a first legal solution: for* $\Sigma = t^a_{a,b}, t^{a,b}_{a,c_1}, t^{c_1}_{c_1,d_1}, t^a_{a,b}, t^b_{b,c_2}, t^{c_2}_{c_2,d_2}$, *which has length 6, we get* $\Sigma \rightsquigarrow \{d_1, d_2\}$. *Notably some, but not all, permutations of* Σ *are also legal solutions; e.g.,* t^b_{b,c_2} *only can be applied as action* t^i *if* $b \in \mathcal{M}^i$. *For one permutation of* Σ, *one application of* $t^a_{a,b}$ *is unnecessary, which leads to the improved solution* $\Sigma' = t^a_{a,b}, t^b_{b,c_2}, t^{c_2}_{c_2,d_2}, t^{a,b}_{a,c_1}, t^{c_1}_{c_1,d_1}$ *of length 5. But the best solution is* $\Sigma^* = t^a_{a,b}, t^b_{b,c_2}, t^{c_2}_{c_2,c_3}, t^{c_3}_{c_3,d_1d_2}$ *of length 4.*

2.2 Design Choices

Next, we highlight some of the design choices made in setting up our model and discuss possible alternatives. For all of them, alternative choices are possible and worth investigating.

Free disposal. By only requiring $\mathcal{U}_{out} \subseteq \mathcal{M}^\ell$ rather than $\mathcal{U}_{out} = \mathcal{M}^\ell$, we are making the assumption that the auctioneer can freely dispose of unwanted goods.

Uniform prices. By defining WINDET in terms of $|\Sigma|$, we are making the implicit assumption that every action is

FEASIBILITY	WINNER DETERMINATION (WINDET)
Instance: an auction instance $\langle \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$.	*Instance:* an auction instance $\langle \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out} \rangle$ and a number $K \in \mathbb{N}$.
Question: is there a legal sequence Σ s.t. $\Sigma \rightsquigarrow \mathcal{U}_{out}$?	*Question:* is there a legal sequence Σ s.t. $\lvert\Sigma\rvert \leqslant K$ and $\Sigma \rightsquigarrow \mathcal{U}_{out}$?

Table 2: Decision Problems for Tool Auctions

equally expensive to execute. Alternatively, one could specify a price for each transformation and ask for the sum of the prices of the actions in Σ to not exceed K.[1]

Unlimited use of transformations. We assume that every transformation can be used any number of times—although the auctioneer has to 'pay' for every single application of it. Alternatively, one could specify for each transformation how often it can be used at most. Going further, one could define a *bidding language* to express constraints on acceptable combinations of actions (Cerquides et al. 2007; Nisan 2006), e.g., to say that the auctioneer has to either purchase all or none of a given set of transformations.

Single-unit *vs.* multi-unit auctions. There are multiple identical copies of each good in G. In particular, $\mathcal{U}_{in}$, $\mathcal{U}_{out}$, and the output bundles of multi-output tools are, in general, multisets. However, we are sometimes going to focus on the special case in which the specification of an auction instance only involves single-unit bundles. We refer to this as the *single-unit restriction*. Observe that, while also under the single-unit restriction the auctioneer might obtain two identical copies of the same good (e.g., by applying the same single-output tool twice), she never *needs* to do so: under the single-unit restriction, for any legal sequence Σ of length $\ell = \lvert\Sigma\rvert$ with $\Sigma \rightsquigarrow \mathcal{U}_{out}$ there exists a legal sequence Σ' with $\lvert\Sigma'\rvert \leqslant \ell$ and $\Sigma' \rightsquigarrow \mathcal{U}_{out}$ for which the auctioneer never has to hold multiple copies of the same good, i.e., for which all $\mathcal{M}^i$ are single-unit bundles as well.

No typing of goods. Tools and ordinary goods are all elements of G. We distinguish them by their *use* within a transformation, not by their essence. Thus, some $x \in G$ could play the role of a tool for one transformation and the role of a good being produced for another transformation. Alternatively, one could have one set of tools and one set of ordinary goods, and only use each for their special purpose. In particular, one could forbid that tools are consumed by tabac tools and thus are unavailable later on in Σ.

For some of our results, we are going to impose this latter restriction, which we refer to as the *nonconsumable-tool restriction*. Formally, it requires that G can be partitioned into G_a and G_b such that (1) for all tabac tools $t^{ab}_{ac} \in \mathcal{T}$ it holds that $a \in G_a$ and $b, c \in G_b$, and (2) all single- and multi-output tools in $\mathcal{T}$ only involve goods in G_a.

[1]Lifting this assumption does not affect the complexity results reported later on in this paper: for WINDET, all hardness results carry over immediately to the more general case (having a cost of 1 is a special case of having arbitrary costs). All membership results can also be extended to the case of arbitrary costs of transformations. In particular, the tractability results of Theorems 12 and 13 can be extended to this case because we can encode arbitrary costs in the reduction to the min-cost network flow problem.

Single tools. We restrict the number of tools required to execute a transformation to 1. Alternatively, one could also permit requiring multiple tools for a single transformation.

Tools only. We restrict attention to scenarios where the only actions available to the auctioneer consist in applying a tool to obtain certain goods. In practice, this should be combined with the option to simply purchase a bundle of goods. By including some default good $\star$ in $\mathcal{U}_{in}$ and ensuring that $\star$ is never consumed by a tabac tool, we can simulate the option of purchasing the bundle B via the multi-output tool $t^{\star}_{\star B}$. Hence, in fact this is not a restriction.

3 Complexity Results

In this section, we present our results on the computational complexity of tool auctions.

These results show that FEASIBILITY is tractable for settings with only single- or only multi-output tools, while WINDET is not. For settings where tabac tools are allowed, even FEASIBILITY is not tractable in general. In fact, it is PSPACE-complete. For the complexity of WINDET in the presence of tabac tools, the encoding of K is relevant. When K is encoded as a binary number, then the problem is PSPACE-complete, while it is NP-complete when K is encoded as a unary number. All of our intractability results apply even under the single-unit restriction. Under the nonconsumable-tool restriction, on the other hand, if only tabac tools are allowed (i.e., if goods are typed), both FEASIBILITY and WINDET are tractable. Refer to Table 3 for an overview of these results.

Transformations	FEASIBILITY	WINDET
single-output tools	in P (Prop 1)	NP-c (Thm 2)
multi-output tools	in P (Prop 3)	NP-c (Thm 4)
tabac tools	PSPACE-c (Prop 7)	PSPACE-c (Thm 8) (K *in binary*)
		NP-c (Thm 9) (K *in unary*)
tabac tools with typed goods	in P (Prop 11)	in P (Thm 12)

Table 3: Overview of Complexity Results

For scenarios where finding a solution is intractable in general, it is often worthwhile to explore whether better results are achievable under the assumption that certain parameters of the problem can be kept small (Downey and Fellows 1999; 2013; Flum and Grohe 2006; Niedermeier 2006; Cygan et al. 2015). Regarding such fixed-parameter-tractability results, the bad news is that the most obvious choice of a parameter, namely the maximum length of the solution $K \geqslant |\Sigma|$, is not a good choice. WINDET for multi-output tools parameterized by K is W[2]-hard and WINDET for tabac tools parameterized by K is W[1]-hard. A better choice for a parameter is the sum of the solution length K and the maximum number of goods that appear in any multi-output tool ($\max_{\langle \mathcal{T},\mathcal{I},\mathcal{O}\rangle \in \mathcal{T}} |\mathcal{O}|$). Another good choice is the size $|\mathcal{U}_{out}|$ of the desired output. For either one of these two parameters, WINDET for multi-output tools is fixed-parameter tractable (FPT). Finally, under the nonconsumable-tool restriction and for the parameter that is the sum of the number of tools which are used in a tabac tool and the number of tools appearing in $\mathcal{U}_{out}$, WINDET is FPT for any combination of our three types of transformations. Table 4 summarises our FPT results.

Transformations	Parameter	Result
single-output tools	$\lvert\Sigma\rvert$	FPT (Cor 6)
multi-output tools	$\lvert\Sigma\rvert$	W[2]-h (Thm 5)
multi-output tools	$\lvert\mathcal{U}_{out}\rvert$	FPT (Cor 6)
multi-output tools	$\lvert\Sigma\rvert + \max \lvert\mathcal{O}\rvert$	FPT (Cor 6)
tabac tools	$\lvert\Sigma\rvert$	W[1]-h (Thm 10)
mixed tools under noncons.-tool restr.	$\lvert\{a \mid t^{ab}_{ac} \in \mathcal{T}\}\rvert + \sum_{a \in G_a} \mathcal{U}_{out}(a)$	FPT (Thm 13)

Table 4: Parameterized Complexity Results for WINDET

3.1 Results for Single-Output Tools

For auction instances that only involve single-output tools, FEASIBILITY is easy, but WINDET is already hard.

Proposition 1. *If all transformations are single-output tools, then* FEASIBILITY *can be decided in polynomial time.*

Proof. We describe a greedy algorithm to construct Σ. We use a multiset $\mathcal{M}$ to keep track of the bundle held by the auctioneer by the end of the prefix of Σ constructed so far. At any given point, $\mathcal{U}_{out} \setminus \mathcal{M}$ is the bundle we still need to obtain. Initialise $\mathcal{M}$ with $\mathcal{U}_{in}$, and Σ with the empty sequence. Then, while there is some t^a_{ab} with $a \in \mathcal{M}$ and $b \notin \mathcal{M}$, add t^a_{ab} to the end of Σ, and add a copy of b to $\mathcal{M}$. After termination, report success if there is no $g \in \mathcal{U}_{out}$ with $g \notin \mathcal{M}$ and failure otherwise. This algorithm is easily seen to be correct and to run in polynomial time (for every element in G we have to inspect every transformation in $\mathcal{T}$ at most once). □

Theorem 2. *If all transformations are single-output tools, then* WINDET *is NP-complete, even under the single-unit restriction.*

Proof (sketch). To show membership in NP, it suffices to see that if a feasible solution exists, there is also a be feasible solution where any good is added at most once to the auctioneer's bundle. Thus only sequences Σ of length at most $|\Sigma| = |G \setminus \mathcal{U}_{in}|$ have to be considered.

Next, we prove NP-hardness under the single-unit restriction by reduction from 3SAT. Let $\varphi = \{c_1, \ldots, c_m\}$ be a 3CNF formula with variables $x_1, x_2, \ldots, x_n$. Without loss of generality, we may assume that φ contains the clauses $(x_i \vee \neg x_i)$ for all $i \in \{1, \ldots, n\}$. We now construct an instance of WINDET. We let $G = \{z\} \cup \{x_i, \neg x_i, c_j \mid 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m\}$, $\mathcal{U}_{in} = \{z\}$, $\mathcal{U}_{out} = \{c_1, c_2, \ldots, c_m\}$, $\mathcal{T} = \mathcal{T}_{\text{lit}} \cup \mathcal{T}_{\text{cl}}$ where $\mathcal{T}_{\text{lit}} = \{t^z_{z,x_i}, t^z_{z,\neg x_i} \mid 1 \leqslant i \leqslant n\}$ and $\mathcal{T}_{\text{cl}} = \{t^{x_i}_{x_i,c_j} \mid x_i \in c_j, c_j \in \varphi\} \cup \{t^{\neg x_i}_{\neg x_i,c_j} \mid \neg x_i \in c_j, c_j \in \varphi\}$. Finally, we let $K = n + m$. It is straightforward to verify that this reduction is correct. □

3.2 Results for Multi-Output Tools

Next, we are going to see that, while the basic complexity results for multi-output tools mirror those for single-output tools, when considering the problem under the parameterized lens, additional sources of complexity become apparent.

Proposition 3. *If all transformations are multi-output tools, then* FEASIBILITY *can be decided in polynomial time.*

Proof (sketch). We can adapt the algorithm given in the proof of Proposition 1. The only difference is that in each round we have to look for a transformation t^a_{aB} with $a \in \mathcal{M}$ and either $B \cap (\mathcal{U}_{out} \setminus \mathcal{M}) \neq \emptyset$ or $B \cap (\{x \in G \mid t^x_{xy} \in \mathcal{T}\} \setminus \mathcal{M}) \neq \emptyset$ (and then add B to $\mathcal{M}$). □

Observe that every single-output tool is a special case of a multi-output tool, so FEASIBILITY for combinations of single- and multi-output tools is polynomial as well.

Theorem 4. *If all transformations are multi-output tools, then* WINDET *is NP-complete, even for the single-unit restriction.*

Proof (sketch). NP-hardness follows from Theorem 2, as single-output tools are also multi-output tools. NP-membership can be shown by adapting the proof of Theorem 2. □

While WINDET is FPT with respect to the length of Σ when only single-output tools are used (see Corollary 6 below), this is unlikely to be the case for multi-output tools.

Theorem 5. *If all transformations are multi-output tools, then* WINDET *is W[2]-hard when parameterized by* $|\Sigma|$. *This remains true even under the single-unit restriction.*

Proof (sketch). We describe an fpt-reduction from the W[2]-complete problem SET COVER. For this problem, inputs consist of triples $(U, \mathcal{S}, k)$, where U is a finite set, $\mathcal{S} \in \mathcal{P}(\mathcal{P}(U))$ is a set of subsets $S_i \subseteq U$ of U, and k is a positive integer. The question is whether there exists a set $\mathcal{S}' \subseteq \mathcal{S}$ of size at most k such that $\bigcup \mathcal{S}' = U$.

Let $(U, \mathcal{S}, k)$ be an arbitrary instance of SET COVER. We construct an instance of WINDET using only multi-output tools. We let $G = \{\star\} \cup U$, $\mathcal{U}_{in} = \{\star\}$, $\mathcal{U}_{out} = G$. For each $S_i \in \mathcal{S}$, we introduce a multi-output tool $t^{\star}_{\star,S_i}$.

The legal sequences Σ of length at most k such that $\Sigma \rightsquigarrow \mathcal{U}_{out}$ are in one-to-one correspondence with the subsets $\mathcal{S}' \subseteq \mathcal{S}$ of size at most k such that $\bigcup \mathcal{S}' = U$. □

The following results follow directly from Theorem 13, which we will establish in Section 3.4.

Corollary 6. *If all transformations are single- or multi-output tools, then* WINDET *is fixed-parameter tractable when parameterized either (1) by $|\mathcal{U}_{out}|$, or (2) by $|\Sigma|$ plus the maximum number of goods that appear in any multi-output tool ($\max_{\langle \tau, \mathcal{I}, \mathcal{O} \rangle \in \mathcal{T}} |\mathcal{O}|$).*

3.3 Results for Tabac Tools

When using (only) tabac tools, solving a tool auction can become highly intractable. This is true even under the single-unit restriction.

Proposition 7. *If all transformations are tabac tools, then* FEASIBILITY *is PSPACE-complete, even under the single-unit restriction.*

Proof (sketch). Membership in PSPACE can be shown routinely, by giving a nondeterministic algorithm that solves the problem in polynomial space. To show PSPACE-hardness, we can use a reduction that has been used to show that propositional planning is PSPACE-hard (Bylander 1994, Thm 3.1), reducing an arbitrary problem in PSPACE to the satisfiability problem of propositional planning. The planning operators used in this reduction correspond exactly to tabac tools. Moreover, the satisfiability problem for propositional planning corresponds exactly to the FEASIBILITY problem. In this reduction, all bundles involved are single-unit bundles. □

The complexity of WINDET depends on how we encode K, the upper bound on the length of permissible solutions.

Theorem 8. *If all transformations are tabac tools, then* WINDET *is PSPACE-complete when K is encoded in binary. This remains true even under the single-unit restriction.*

Proof (sketch). Encoding K in binary allows us to express an exponential bound on the length of solutions. Therefore, the PSPACE-hardness proof of Proposition 7 also works for WINDET. Membership in PSPACE can be shown entirely analogously to the case of FEASIBILITY. □

Theorem 9. *If all transformations are tabac tools, then* WINDET *is NP-complete when K is encoded in unary. This remains true even under the single-unit restriction.*

Proof (sketch). Membership in NP follows from the fact that when K is encoded in unary, we can guess a sequence Σ in polynomial time.

To show NP-hardness, we describe a reduction from WINDET for the case where all transformations are single-output tools (cf. Theorem 2). Let $(G, \mathcal{T}, \mathcal{U}_{in}, \mathcal{U}_{out}, K)$ be an arbitrary instance of WINDET where all $t \in \mathcal{T}$ are single-output tools. To G we add K new goods $h_1, \dots, h_K$. Each transformation $t^{a}_{ab} \in \mathcal{T}$ we replace by K tabac tools $t^{ah_1}_{ab}, \dots, t^{ah_K}_{ab}$. To $\mathcal{U}_{in}$, we add the goods $h_1, \dots, h_K$, and $\mathcal{U}_{out}$, and K we leave unchanged.

We omit a detailed proof of the fact that there exists a sequence Σ of length at most K such that $\Sigma \rightsquigarrow \mathcal{U}_{out}$ for the original instance if and only if such a sequence exists for the newly constructed instance. □

Theorem 10. *If all transformations are tabac tools, then* WINDET *is W[1]-hard when parameterized by K, even under the single-unit restriction with $|\mathcal{U}_{out}| = 1$.*

Proof (sketch). We describe an fpt-reduction from the W[1]-complete MULTICOLOURED CLIQUE (Fellows et al. 2009), which asks, given an undirected graph $\mathcal{G} = (V, E)$ the vertices of which are coloured with k colours, whether there is a clique in $\mathcal{G}$ containing one vertex from each colour class (a *multi-coloured clique*). The parameter for this problem is k.

Given an instance of MULTICOLOURED CLIQUE with $\mathcal{G} = (V, E)$ where the vertex set V is a disjoint union of k colour classes $V_1 \,\dot\cup\, \cdots \,\dot\cup\, V_k$, and $k \in \mathbb{N}$, we construct an instance of WINDET as follows (we assume $k \geqslant 3$, otherwise an easier construction is possible). The set of goods is

$$G := \{\star\} \cup \{x_e \mid e \in E\} \cup \{z_e \mid e \in E\} \cup Y$$

with

$$Y := \{y_{i,j,\ell} \mid 1 \leqslant i \leqslant k, 1 \leqslant j < \ell \leqslant k, i \neq j, i \neq \ell\},$$

$\mathcal{U}_{in} = \{\star\}$, and $\mathcal{U}_{out} = Y$. The number of goods in the output is $|Y| = k\binom{k-1}{2}$. For each $e \in E$, we introduce transformations $t^{\star}_{\star\, x_e}$ and $t^{x_e}_{x_e\, z_e}$, where the first transformation allows production of goods x_e and the second one allows production of goods z_e using x_e as a tool. Note that these transformations are single-output tools, but they can be transformed into tabac tools as done in the proof of Theorem 9. For each pair $e_1 = \{v_i, v_j\}$, $e_2 = \{v_i, v_\ell\} \in E$, with $e_1 \neq e_2$, $v_j \in V_j$, and $v_\ell \in V_\ell$, we create a tabac tool $t^{x_{e_1} z_{e_2}}_{x_{e_1} y_{i,j,\ell}}$. To put it into words, the tabac tool $t^{x_{e_1} z_{e_2}}_{x_{e_1} y_{i,j,\ell}}$ allows transforming z_{e_2} into a good $y_{i,j,\ell}$, using x_{e_1} as a tool, where e_1 and e_2 share a vertex in color class i, and have differing vertices in color classes j and ℓ, respectively. We let $K = \binom{k}{2} + 2k\binom{k-1}{2}$. The only way to produce all goods $y_{i,j,\ell}$ within the budget K is to first produce the goods x_e for all edges e in a multi-coloured clique, and then to use them as tools to produce the goods $y_{i,j,\ell}$ (with goods z_e as intermediate step).

If $\mathcal{G}$ has a multicoloured clique of size k, we produce those goods corresponding to the $\binom{k}{2}$ edges of the clique by the single-output tools, i.e., we apply the transformation $t^{\star}_{\star\, x_e}$ for each edge e of the clique. Since there is only one vertex of each colour class in the clique, we can produce in turn all output goods Y with the tabac tools given that we produce the good consumed in each step prior to the application of the tabac tool (which is possible since we have produced all necessary tools before). Hence, we end up with a legal sequence Σ of length $\binom{k}{2} + 2k\binom{k-1}{2}$ achieving $\mathcal{U}_{out}$.

Conversely, it can be checked easily that it is impossible to produce the entire set Y as an output in $\binom{k}{2} + 2k\binom{k-1}{2}$ steps if there is no multicoloured clique of size k in $\mathcal{G}$.

To achieve an output of size 1, let $k' := \binom{k-1}{2}k$ and rename the elements of Y as $y_1, \dots, y_{k'}$; we create $k' - 1$ additional goods $y_{(1,2)}, \dots, y_{(1,\dots,k')}$ and the $k' - 1$ tabac tools $t^{y_{(1,\dots,i)}\, y_{i+1}}_{y_{(1,\dots,i)}\, y_{(1,\dots,i+1)}}$, where $y_{(1)} := y_1$. Then the new output set $\mathcal{U}_{out}$ consists of the single element $y_{(1,\dots,k')}$. To account for this modification of the instance, we have to adapt the parameter to $K = \binom{k}{2} + 2k\binom{k-1}{2} + k' - 1$.

Also note that converting the single-output tools into tabac tools introduces new elements in $\mathcal{U}_{in}$. In the same notation as in the proof of Theorem 9 this introduces input goods $h_1, \dots, h_{k''}$, i.e., $\mathcal{U}_{in} = \{\star, h_1, \dots, h_{k''}\}$. Here, we have $k'' = \binom{k}{2} + k\binom{k-1}{2}$ since we need one such good for each application of a single-output tool. $\square$

3.4 Results for the Nonconsumable Tools

Next, we consider auction scenarios that satisfy the nonconsumable-tool restriction (see Section 2.2). As we shall see, under this restriction we can achieve particularly positive results. Intuitively, the nonconsumable-tool restriction makes both FEASIBILITY and WINDET easier to solve, because we can separate the application of tabac tools from the application of other transformations.

Indeed, as we shall see next, in such cases we can often solve auctions by reduction to a *minimum-cost network flow problem* (Ahuja, Magnanti, and Orlin 1993). Inputs for this problem consist of a directed graph $\mathcal{G} = (V, E)$ together with a positive integer $r \in \mathbb{N}$. The graph $\mathcal{G}$ has a dedicated source node $s \in V$ (with only outgoing edges) and a dedicated target node $t \in V$ (with only incoming edges). Moreover, each edge $e \in E$ is associated with a cost $c_e \in \mathbb{N}$ and a capacity $u_e \in \mathbb{N}$. A *network flow* for $\mathcal{G}$ is a mapping $\mu\colon E \to \mathbb{N}$ such that for each $e \in E$ it holds that $\mu(e) \leqslant u_e$ and for each $v \in V \setminus \{s, t\}$ it holds that $\sum_{e=(u,v)\in E} \mu(e) = \sum_{e=(v,u)\in E} \mu(e)$. We say that the *value* of a network flow μ is $\sum_{e=(u,t)\in E} \mu(e)$, and the *cost* of μ is $\sum_{e\in E} c_e\mu(e)$. The problem consists of finding a network flow of $\mathcal{G}$ of value r that has minimum cost.

Recall that G_b is the set of goods consumed and produced by tabac tools, and G_a is the set of all other goods.

Proposition 11. *Under the nonconsumable-tool restriction, if all transformations are single-output, multi-output, or tabac tools,* FEASIBILITY *can be decided in polynomial time.*

Proof (sketch). First, we compute the maximum set $G'_a \subseteq G_a$ of goods among G_a that we can obtain from $\mathcal{U}_{in}$ using transformations that are single- or multi-output tools, similarly to the algorithm described in the proof of Proposition 3. Then, we discard all tabac tools t^{ab}_{ac} for which $a \notin G'_a$. We then decide whether we can produce the desired quantity of goods in G_b (according to $\mathcal{U}_{out}$) from the quantities of goods in G_b that are available in $\mathcal{U}_{in}$. We do so by reducing the problem to an instance of the network flow problem.

We create a network containing a dedicated source node s, a dedicated target node t, and a node b for each good $b \in G_b$. For each $b \in G_b$, we add an arc from s to b with capacity $\mathcal{U}_{in}(b)$, and an arc from b to t with capacity $\mathcal{U}_{out}(b)$. Moreover, for each undiscarded tabac tool $t^{ab}_{ac} \in \mathcal{T}$, we add an arc from b to c with unlimited capacity. There is a network flow from s to t of value $\sum_{b\in G_b} \mathcal{U}_{out}(b)$ in this network if and only if there is a legal sequence Σ such that $\Sigma \rightsquigarrow \mathcal{U}_{out}$. Since network flow is solvable in polynomial time (see, e.g., Ahuja, Magnanti, and Orlin 1993), we know that FEASIBILITY is as well. Existence of an integral solution is guaranteed as well, since all capacities are integral. $\square$

For WINDET, we cannot hope for a similarly positive result, given that it already is NP-complete for single-output tools alone (cf. Theorem 2), a scenario in which the nonconsumable-tool restriction does not add any further constraints. However, when only tabac tools are used, we can still achieve a significant improvement over Theorems 9 and 10.

Theorem 12. *Under the nonconsumable-tool restriction, if all transformations are tabac tools,* WINDET *can be decided in polynomial time.*

Proof (sketch). We can use a similar construction as for the second part of the proof of Proposition 11. Now, to reduce the problem to the minimum-cost network flow problem, we simply assign unit cost to all arcs in the network (except for the arcs from s and the arcs to t, to which we assign cost 0). $\square$

As argued earlier, if we combine all three types of tools, WINDET is intractable, even under the nonconsumable-tool restriction. However, if we restrict the size of the problem appropriately, then we can construct an efficient algorithm even for this combined setting.

Theorem 13. *Under the nonconsumable-tool restriction, if all transformations are single-output, multi-output, or tabac tools,* WINDET *is fixed-parameter tractable when parameterized by (the sum of) the number k_1 of different goods in G_a that appear in some tabac tool, and the total number k_2 of goods in G_a that appear in $\mathcal{U}_{out}$, i.e., $k_1 = |G_{a,\text{tabac}}|$, where $G_{a,\text{tabac}} = \{a \mid t^{ab}_{ac} \in \mathcal{T}\}$, and $k_2 = \sum_{a\in G_a} \mathcal{U}_{out}(a)$.*

Proof. Take an arbitrary instance of WINDET that satisfies the condition from the statement above. Let $\mathcal{U}_{out,a}$ denote the multiset $\mathcal{U}_{out}$ restricted to goods in G_a, i.e., $\mathcal{U}_{out,a}(g_a) = \mathcal{U}_{out}(g_a)$ for all $g_a \in G_a$ and $\mathcal{U}_{out,a}(g_b) = 0$ for all $g_b \in G_b$. Moreover, let $\mathcal{T}_{\text{tabac}}$ denote the set of all tabac tools $t^{ab}_{ac} \in \mathcal{T}$. The algorithm proceeds in several stages.

In the first stage, we use dynamic programming to compute a table with entries $D(s, G_1, G_2, m)$, for each $s \in G_a$, each $G_1 \subseteq \mathcal{U}_{out,a}$, each $G_2 \subseteq G_{a,\text{tabac}}$, and each $1 \leqslant m \leqslant |G_a|$. Each such table entry will contain 'yes' or 'no', depending on whether there exists a sequence Σ of length at most m that is legal for the bundle $\mathcal{U}_{in,s} = \{s\}$ and achieves (from $\mathcal{U}_{in,s}$) some bundle $\mathcal{M}$ with $G_1 \subseteq \mathcal{M}$ and $G_2 \subseteq \mathcal{M}$. One could additionally store a witnessing sequence Σ for each entry containing 'yes.' The size of this table is bounded by $2^{k_1+k_2} \cdot \text{poly}(n)$ (where n denotes the input size), and we can fill its entries, by induction on m, in time polynomial in the size of the table and in the input size n, using a dynamic programming approach.

In the second stage, we use the table we computed in the first stage to fill a table with entries $D(G_1, G_2, m)$, for each $G_1 \subseteq \mathcal{U}_{out,a}$, and each $G_2 \subseteq G_{a,\text{tabac}}$. Each such entry $D(G_1, G_2, m)$ indicates whether there exists a sequence Σ of length at most m that is legal for $\mathcal{U}_{in}$ and achieves some bundle $\mathcal{M}$ with $G_1 \subseteq \mathcal{M}$ and $G_2 \subseteq \mathcal{M}$. Again, one could additionally store a witness for each entry containing 'yes' and filling the entries can be done in polynomial time, using a dynamic programming approach.

Then, in the third stage, we nondeterministically guess a subset $G'_{a,\text{tabac}} \subseteq G_{a,\text{tabac}}$ of goods in G_a that will occur in the tabac tools t^{ab}_{ac} appearing in sequence Σ achieving $\mathcal{U}_{out}$. (Since there are at most 2^{k_1} such sets $G'_{a,\text{tabac}}$, we can make the algorithm deterministic by simply iterating over all possibilities.) Let $\mathcal{T}'_{\text{tabac}} \subseteq \mathcal{T}_{\text{tabac}}$ denote the set of tabac tools t^{ab}_{ac} such that $a \in G'_{a,\text{tabac}}$. We then find the minimum length ℓ such that the table entry $D(\mathcal{U}_{out,a}, G'_{a,\text{tabac}}, \ell)$ contains 'yes'.

In the fourth, and final, stage of the algorithm, we determine whether we can extend the sequence Σ_0 of length ℓ that achieves a bundle including both $\mathcal{U}_{out,a}$ and $G'_{a,\text{tabac}}$ to a sequence Σ of length at most K that achieves $\mathcal{U}_{out}$. We only need to consider (multiple applications of) transformations in $\mathcal{T}'_{\text{tabac}}$ to append to Σ_0 in order to obtain Σ. We encode the question whether we can obtain a suitable Σ by appending at most $K - \ell$ transformations to Σ_0 as an instance of the minimum-cost network flow problem.

This encoding is exactly the encoding used in the proof of Theorem 12. For this encoding, we use $\mathcal{U}_{in} + \mathcal{U}_{out,a} + G'_{a,\text{tabac}}$ as input state. Moreover, we discard all single- and multi-output tools, as well as all tabac tools t^{ab}_{ac} for which $a \notin G'_{a,\text{tabac}}$. Finally, as upper bound on the length of the sequence Σ_1 such that $\Sigma = \Sigma_0\Sigma_1$, we use $K - \ell$. This encoding can easily be done in polynomial time, and the constructed instance of minimum-cost network flow can be solved in polynomial time (see, e.g., Ahuja, Magnanti, and Orlin 1993).

We can straightforwardly modify this algorithm to return a legal sequence Σ such that $\Sigma \rightsquigarrow \mathcal{U}_{out}$, if it exists. $\square$

4 Related Work

Our model of tool auctions follows prior work on market mechanisms for enabling automatic *supply chain formation* (Walsh, Wellman, and Ygge 2000; Walsh and Wellman 2003; Babaioff and Nisan 2004; Cerquides et al. 2007), even if the distinction between tools and ordinary goods has been absent from those earlier contributions. The model of Walsh and Wellman (2003), in particular, is similar to ours in that they also impose strong qualitative constraints on transformations. Their transformations have either exactly one output good (so-called 'producers') or no output goods at all (so-called 'consumers'). Unlike us, they however also impose a strong structural constraint and require the graph defined by the set of transformations to be acyclic. Walsh and Wellman do not analyse the complexity of their model.

Cerquides et al. (2007) show that WINDET is NP-complete for *mixed auctions*. Tool auctions are a restricted form of mixed auctions—except for the fact that we permit unlimited use of transformations (see Section 2.2). This difference is crucial: Proposition 7, establishing PSPACE-completeness, holds only because a solution sequence could be exponentially long relative to the number of transformations available. In particular, this result does not contradict the NP-membership result of Cerquides et al. (2007).

Fionda and Greco (2013) analyse the complexity of mixed auctions in great depth. Part of their work focuses on *structural* restrictions, such as acyclicity, which we have not considered here. Structural restrictions are of interest when there is empirical evidence that they may be satisfied for typical real-world problem instances. OInstead, our focus has been on *qualitative* restrictions, which can be imposed on each transformation in isolation and thus can simply be enforced in practice, by appropriately restricting the protocol used for communicating bids. Fionda and Greco also establish important results on qualitative restrictions, which however are largely negative. In essence, they show that transformations with a single input and a single output good give rise to tractable problems, while even only allowing a second good at either the input or the output side makes things intractable. Our results show that better results are achievable by introducing the notion of 'tool' and using it to impose qualitative constraints between input and output goods.

The idea of trying to identify '*tractability islands*' for solving combinatorial auctions, i.e., restrictions to the general setting that are both of practical interest for modelling application scenarios and that allow for optimal solutions that can be computed efficiently, goes back to the work of Rothkopf, Pekeč, and Harstad (1998) and has been explored by several other authors since then (Conitzer, Derryberry, and Sandholm 2004; Müller 2006; Gottlob and Greco 2007; Döcker et al. 2016).

Finally, there is a large body of work on other kinds of *combinatorial auctions* (Cramton, Shoham, and Steinberg 2006), which we do not review here. We emphasise, however, that *combinatorial exchanges* (Sandholm et al. 2002) differ from both tool auctions and the other auction models discussed here in that they do not require feasible solutions to correspond to a *sequence* of transformations. Instead, they only require the totality of all output goods to subsume the totality of all input goods.

5 Conclusion

We have introduced tool auctions and provided a detailed complexity analysis of both the feasibility and the winner determination problem of such auctions. With the exception of the feasibility problem for single- and multi-output tools, these problems are intractable in general. However, for the case of tabac tools, we have identified a special case of practical interest where they are polynomial-time solvable. In addition, we have analysed the winner determination problem from a parameterized complexity perspective and identified both tractable and intractable scenarios.

A first direction for future work concerns the development of practical algorithms for solving tool auctions. Our (fixed-parameter) tractability results point the way for the design of efficient algorithms for certain special cases. For cases that are intractable, techniques such as combinatorial optimisation and heuristic-guided search should be explored. A natural starting point is prior work on such methods for mixed auctions (Giovannucci et al. 2008; Ottens and Endriss 2008) and other generalisations of the basic combinatorial auction model (Sandholm et al. 2002).

A second direction concerns the analysis of real-world instances of our general model to get a clearer picture of what parameters can be assumed to typically take small values in practice. Besides industrial supply-chain management, possible applications include online freelancing platforms.

Acknowledgments. This work was partly supported by COST Action IC1205 on Computational Social Choice. RdH was furthermore supported by the Austrian Science Fund (FWF), project J4047.

References

Ahuja, R. K.; Magnanti, T. L.; and Orlin, J. B. 1993. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall.

Babaioff, M., and Nisan, N. 2004. Concurrent auctions across the supply chain. *Journal of Artificial Intelligence Research (JAIR)* 21:595–629.

Bylander, T. 1994. The computational complexity of propositional STRIPS planning. *Artificial Intelligence* 69(1):165–204.

Cerquides, J.; Endriss, U.; Giovannucci, A.; and Rodríguez-Aguilar, J. A. 2007. Bidding languages and winner determination for mixed multi-unit combinatorial auctions. In *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI)*.

Conitzer, V.; Derryberry, J.; and Sandholm, T. W. 2004. Combinatorial auctions with structured item graphs. In *Proc. 19th National Conference on Artificial Intelligence (AAAI)*.

Cramton, P.; Shoham, Y.; and Steinberg, R., eds. 2006. *Combinatorial Auctions*. MIT Press.

Cygan, M.; Fomin, F. V.; Kowalik, L.; Lokshtanov, D.; Marx, D.; Pilipczuk, M.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer.

Döcker, J.; Dorn, B.; Endriss, U.; and Krüger, D. 2016. Complexity and tractability islands for combinatorial auctions on discrete intervals with gaps. In *Proc. 22nd European Conference on Artificial Intelligence (ECAI)*. IOS Press.

Downey, R. G., and Fellows, M. R. 1999. *Parameterized Complexity*. Springer.

Downey, R. G., and Fellows, M. R. 2013. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer.

Fellows, M. R.; Hermelin, D.; Rosamond, F. A.; and Vialette, S. 2009. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science* 410(1):53–61.

Fionda, V., and Greco, G. 2013. The complexity of mixed multi-unit combinatorial auctions: Tractability under structural and qualitative restrictions. *Artificial Intelligence* 196:1–25.

Flum, J., and Grohe, M. 2006. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. Springer.

Giovannucci, A.; Vinyals, M.; Rodríguez-Aguilar, J. A.; and Cerquides, J. 2008. Computationally-efficient winner determination for mixed multi-unit combinatorial auctions. In *Proc. 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Gottlob, G., and Greco, G. 2007. On the complexity of combinatorial auctions: Structured item graphs and hypertree decomposition. In *Proc. 8th ACM Conference on Electronic Commerce (EC)*. ACM.

Müller, R. 2006. Tractable cases of the winner determination problem. In Cramton, P.; Shoham, Y.; and Steinberg, R., eds., *Combinatorial Auctions*. MIT Press.

Niedermeier, R. 2006. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press.

Nisan, N. 2006. Bidding languages for combinatorial auctions. In Cramton, P.; Shoham, Y.; and Steinberg, R., eds., *Combinatorial Auctions*. MIT Press.

Ottens, B., and Endriss, U. 2008. Comparing winner determination algorithms for mixed multi-unit combinatorial auctions. In *Proc. 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Rothkopf, M. H.; Pekeč, A.; and Harstad, R. M. 1998. Computationally manageable combinational auctions. *Management Science* 44(8):1131–1147.

Sandholm, T. W.; Suri, S.; Gilpin, A.; and Levine, D. 2002. Winner determination in combinatorial auction generalizations. In *Proc. 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Walsh, W. E., and Wellman, M. P. 2003. Decentralized supply chain formation: A market protocol and competitive equilibrium analysis. *Journal of Artificial Intelligence Research (JAIR)* 19:513–567.

Walsh, W. E.; Wellman, M. P.; and Ygge, F. 2000. Combinatorial auctions for supply chain formation. In *Proc. 2nd ACM Conference on Electronic Commerce (EC)*. ACM.