Control of Fair Division

Haris Aziz
Data61 and UNSW
Australia
haris.aziz@data61.csiro.au

Ildikó Schlotter
Budapest University of
Technology and Economics
ildi@cs.bme.hu

Toby Walsh
Data61 and UNSW
Australia
toby.walsh@data61.csiro.au

Abstract

We initiate the study of control actions in fair division problems where a benevolent or malicious central organizer changes the structure of the fair division problem for self-interest or to benefit one, some or all agents. One motivation for such control is to improve fairness by minimally changing the problem. As a case study, we consider the problem of adding or deleting a small number of items to improve fairness. For two agents, we present polynomial-time algorithms for adding or deleting the minimum number of items to achieve ordinal envy-freeness. For three agents, we show that both problems, as well as the more basic problem of checking whether an envy-free allocation exists, are NP-complete. This closes a problem open for over five years. Our framework leads to a number of interesting directions in the area of fair division.

1 Introduction

When allocating resources to agents, a basic and widely sought after requirement is fairness [Bouveret *et al.*, 2016; Brams and Taylor, 1996; Moulin, 2003]. Fairness has been formalized in a number of ways such as envy-freeness, proportionality, max-min fair share. A fundamental problem with indivisible items is that a fair allocation may not exist. However, with small changes to the given instance, a fair allocation might exist. This can also be viewed as the minimal compromise required to achieve fairness. We pursue this thought by considering control in fair division. More generally, we identify various control actions that a central organizer may use to benefit himself or herself, benefit or harm other agents, or simply to meet some goal.

We will typically assume that the chair has full knowledge of the ordinal preferences of the different agents. This may be appropriate for several reasons. First, this models the situation where the chair collects preferences, and then runs a fair division algorithm. This is the case in allocating courses at the Harvard Business School. Second, this is a special case of partial information. It therefore provides a lower bound on the complexity in the presence of partial information. If a problem is computationally intractable with complete information, then it is at least as hard with incomplete.

Contributions: We propose the study of control actions in fair division problems. This aligns with the direction proposed by Bartholdi, III *et al.* [1992] who initiated the computational study of control actions in voting. As a case study, we focus on adding or deleting a few items to ensure that an envy-free allocation exists. For the case of two agents, we present polynomial-time algorithms for adding and deleting the minimum number of items so as to ensure that an ordinal envy-free allocation exists. For the case of three agents, we show that both problems are NP-complete, as well as the more basic problem of checking whether there exists an ordinal envy-free allocation. The latter problem was previously open not just for three agents, but in fact for any constant number of agents [Aziz *et al.*, 2015b; Bouveret *et al.*, 2010].

2 Related Work

When fairness cannot be achieved, one approach is to relax the fairness notion by using approximation [Procaccia and Tennenholtz, 2013; Procaccia and Wang, 2014]. In this paper, we consider problems in which we do not relax the fairness concept but relax the problem by adding or deleting items to ensure fairness. In recent work, Nguyen and Vohra [2014] showed that for the problem of stable matching with couples, perturbing the capacities of schools results in an instance with a stable matching. There is also related work where certain items are duplicated in housing markets [Cechlárová and Schlotter, 2010]. Segal-Halevi *et al.* [2015] considered the idea of not allocating all the divisible resource not for the sake of achieving fairness but in order to obtain faster protocols for allocating cake in a fair manner. Finally, we present results on envy-freeness when agents have preferences over individual items. The notion is equivalent to itemwise envy-freeness by Brams et al., necessary envy-freeness as defined by Bouveret *et al.* [2010] or SD envy-freeness by Aziz *et al.* [2014].

3 Control of Fair Division

The control actions available to the chair are similar to those studied in voting [Bartholdi, III *et al.*, 1992]. There are also some new control actions specific to fair division.

Item addition/deletion/replacement: The chair might add, delete or replace items. For example, can the chair ensure the existence of a fair allocation by donating some additional items?

Agent addition/deletion/replacement: The chair might add, delete or replace agents. For example, in the FoodBank local problem [Aleksandrov *et al.*, 2015a], can we introduce a new charity without lowering greatly the egalitarian social welfare?

Item/agent partitioning: The chair might partition the items and/or agents into disjoint sets. For example, when allocating rooms in St John's College at the University of Cambridge, the (typically more desired) rooms in the College are allocated before the rooms outside of the College, and first year students are considered before second year students.

One possible goal of the chair might be to improve fairness. For instance, with indivisible goods, an envy-free allocation may not exist (consider two agents and a single good). However, the chair might be able to add or delete a small number of items to ensure envy-freeness. This gives rise to a number of natural computational questions. For instance, can we delete k or fewer items so that an envy-free allocation exists? Similar questions can be asked to ensure other fairness properties like proportionality, max-min fair share, etc. In general, if it is computationally intractable to check if an allocation exists with a given property, then adding or deleting items to ensure this property is also computationally intractable.

Observation 1. *If checking whether an allocation exists with property Φ is NP-hard, then adding/deleting/replacing the minimum number of items to ensure Φ is also NP-hard.*

For example, it is NP-complete to check if every agent can receive the max-min fair share. Hence, it is NP-hard to add the minimum number of items to ensure a max-min fair share allocation exists. We might also consider control actions to achieve other goals (e.g. a minimum egalitarian welfare). Given a particular mechanism, we might also use control actions to achieve a particular outcome (e.g. that a given agent gets a certain item or set of items).

4 Envy-freeness

We focus now on the well-known fairness property of *envy-freeness (EF)*. Formally, the input of the ENVY-FREENESS problem can be described as a triple (N, I, L), where N is a set of agents, I a set of indivisible items, and L is a collection of preference lists L^A for each agent $A \in N$. Each preference list L^A is a strict linear ordering over the set I of items. For a linear ordering $L = (s_1, \ldots, s_m)$ over a set $S = \cup_{i=1}^m s_i$ of items, we let $L(i : j) = (s_i, s_{i+1}, \ldots, s_j)$ for any $1 \le i \le j \le m$. For $X \subseteq S$, we let $L_{|X}$ be the restriction of L to X, and write $[L_{|X}]$ for the set of elements in $L_{|X}$.

When there are only two agents, we will denote them by A and B. An assignment π of items to agents is an *allocation*, and π is *complete* if it assigns each item of I to some agent.

When reasoning about preferences over bundles of items, an agent may be required to express preferences over an exponential number of bundles. A compact way of expressing preferences over bundles is for agents to express preferences over individual items and then extend them over bundles of items with respect to the *responsive set extension*. In this notion, we say that an agent A *prefers* a set I_1 of items over a set I_2 of items if there exists an injection f from I_2 to I_1 such that for each item $x \in I_2$, agent A prefers the item $f(x)$ over x. An allocation is *(itemwise) envy-free* if each agent prefers its own set of items over any set of items allocated to some other agent. In the ENVY-FREENESS problem, the task is to find a complete envy-free allocation.

Example 1. *Suppose agents A and B have the following preferences over items $1, 2, 3, 4$.*

$$A: \quad 1 \succ 2 \succ 3 \succ 4$$
$$B: \quad 2 \succ 1 \succ 4 \succ 3$$

In that case, the unique itemwise envy-free allocation is one in which A gets 1 and 3, while agent B gets 2 and 4.

An immediate result of Observ. 1 is that, when the number of agents is not bounded, adding/deleting/replacing items to ensure envy-freeness is NP-hard, because finding an envy-free allocation is NP-hard in this case [Bouveret *et al.*, 2010].

Theorem 1. *The problems of adding/deleting/replacing items to ensure envy-freeness are NP-hard to decide.*

4.1 Two Agents

With two agents, the problem of deleting the fewest items to ensure envy-freeness is solved by Brams, Kilgour and Klamler's AL mechanism. Thus, with two agents, deciding if we can delete k items to ensure envy-freeness takes linear time. Theorem 3 of Brams *et al.* [2014] states that the AL mechanism returns a maximal envy-free allocation. Though it is clear that their algorithm returns an envy-free allocation, their reasoning does not in fact prove that it allocates the maximum number of items possible in any envy-free allocation. For completeness, we prove that AL indeed satisfies this property.

Notation: Before proceeding with our technical results, let us introduce some notation, some of it based on the paper [Brams *et al.*, 2014]. For some subset $X \subseteq I$ of items, we may compute those indices i for which $[L^A_{|X}(1 : i)] = [L^B_{|X}(1 : i)]$; let $i_1, i_2, \ldots, i_s$ denote these indices in an increasing order (observe that $i_s = |X|$ must hold), and we set $i_0 = 0$. We define the *equality segments* $S_1, \ldots, S_s$ for X by letting the segment S_j equal the set $[L^A_{|X}(i_{j-1} + 1 : i_j)] = [L^B_{|X}(i_{j-1} + 1 : i_j)]$, for each $j = 1, \ldots, s$. In Example 1 above, the equality segments of the item set are $\{1, 2\}$ and $\{3, 4\}$. Notice that any item set $X' \subseteq X$ is an equality segment for X if and only if there are indices ℓ and k, with $1 \le \ell \le k \le |X|$ such that $X' = [L^A_{|X}(\ell : k)] = [L^B_{|X}(\ell : k)]$, and X is inclusion-wise minimal with respect to this property.

We will make heavy use of the following characterization proposed by Brams *et al.* [2014]: there is an envy-free allocation of the items of $X \subseteq I$ to agents A and B if the condition $[L^A_{|X}(1 : i)] = [L^B_{|X}(1 : i)]$ (called condition C_i in [Brams *et al.*, 2014]) holds only for even values of i, or equivalently, if each equality segment for X has even size.

Theorem 2. *With two agents, deleting a minimum number of items to ensure envy-freeness can be decided in time linear in the number of items.*

Proof. We begin by describing algorithm AL. It repeats the following step until all items have been allocated to agent A, B, or placed in the contested pile C. We will refer to an item as *unprocessed* if it has not been allocated to A or B, or placed in C. If the most preferred unprocessed item differs for agents A and B, then each agent picks its most preferred item. Otherwise, if the most preferred unprocessed item o coincides, then we check whether we can give it to agent A: if the partial assignment where o is given to agent A while B's next most preferred unprocessed item is given to B still satisfies envy-freeness, then we allow such an allocation. If not, we check in the same way whether we can give o to agent B. If o cannot be given to either agent, we put it in C.

Let us consider a point during the running of AL when some item x is placed in the contested pile C. Let S_A and S_B be the set of items allocated so far to A and B, respectively. By definition of AL, (1) each agent prefers items already allocated to it over x, (2) x is the most preferred item among all unprocessed items for both agents and (3) x cannot be allocated to any of the agents while allocating another unprocessed item to the other agent without causing envy. This implies that (4) there is no unprocessed item that either agent prefers to any item in $S_A \cup S_B \cup \{x\}$. To see this, assume for contradiction that some agent, say B, prefers an unprocessed item y over an item $z \in S_A \cup S_B \cup \{x\}$. By (1) and (2), we get $z \in S_A$. However, in this case allocating x to A and y to B would still yield an envy-free allocation. Indeed, given a bijection $f : S_A \to S_B$ showing that B is not envious of A, we can create a bijection $g : S_A \cup \{x\} \to S_B \cup \{y\}$ proving the envy-freeness of the extended allocation by setting $g(z) = y$, $g(x) = f(z)$, and $g(o) = f(o)$ for all items $o \in S_A \setminus \{z\}$.

Let P_i be the set of all items processed up to the point when the i-th item x_i is placed in C (including x_i itself). Claim (4) implies that for each i, the items in P_i occupy the top k_i positions both in L^A and L^B for some k_i. Hence, the set $P_i \setminus P_{i-1}$ (where we set $P_0 = \emptyset$) must be the union of equality segments, and since $P_i \setminus P_{i-1}$ consists of an even set of items allocated to the agents by AL plus the item x_i, we know that at least one of these equality segments must be odd. Hence, any envy-free allocation must leave at least one item in $P_i \setminus P_{i-1}$ unallocated, proving that AL allocates the maximum number of items possible in any envy-free allocation. □

Similarly, with two agents, we can compute the fewest number of items to add to ensure envy-freeness in polynomial time. In this setting, we assume that I is partitioned into a set F of *fixed* items and a set E of *eligible* items. The task is to find a subset $E' \subseteq E$ of minimum size such that $F \cup E'$ admits an envy-free allocation (with respect to the preference lists $L^A_{|F \cup E'}$ and $L^B_{|F \cup E'}$), or report if no such set exists.

Theorem 3. *With two agents, the problem of adding a minimum number of items to ensure envy-freeness can be decided in time polynomial in the number of items.*

Proof. Let $F_1, \ldots, F_s$ be the equality segments for the set F of fixed items. Let us fix some eligible item e. W.l.o.g., we may assume that e appears at a higher position in $L^A_{|F \cup \{e\}}$ than in $L^B_{|F \cup \{e\}}$; otherwise, we can swap the roles of A and B in the following definition. We say that e *starts* at the equality segment F_i, if F_i is the first segment such that agent A prefers e to its least preferred item in F_i. Analogously, e *ends* at the equality segment F_j, if F_j is the last segment such that agent B prefers its most preferred item in F_j to e.

The following key observation describes how the addition of an item to an instance alters its equality segments.

Proposition 1. *Suppose that the equality segments for a set X of items are $F_1, \ldots, F_s$, and some item $e \in I \setminus X$ starts at F_i and ends at F_j. If $j \geq i$, then the equality segments for $X \cup \{e\}$ are $F_1, \ldots, F_{i-1}, \cup_{h=i}^{j} F_h \cup \{e\}, F_{j+1}, \ldots, F_s$. If $j < i$, then $j = i - 1$, and the equality segments for $X \cup \{e\}$ are $F_1, \ldots, F_j, \{e\}, F_i, \ldots, F_s$.*

Thus, adding an eligible item e to the set of items merges all segments in between its starting and ending segment into one new equality segment, containing also e in addition.

Using Prop. 1, we are going to compute a solution E' by dynamic programming. Let us define a set $M(i, j)$ for each pair of indices i and j with $1 \leq i \leq j \leq s$ as a smallest possible subset of the set E of eligible items whose addition to F merges the equality segments $F_i, \ldots, F_j$ while not altering the remaining segments. That is, $M(i, j)$ is a set $X \subseteq E$ of minimum cardinality such that the equality segments for $F \cup X$ are exactly $F_1, \ldots, F_{i-1}, \cup_{h=i}^{j} F_h \cup X, F_{j+1}, \ldots, F_s$. If there is no such set, we set $M(i, j) = \star$.

After determining the equality segments, we compute the sets $M(i, j)$ in polynomial time as follows. Initially we set $M(i, j) = \star$ for each i and j. Then we take each index $j = 1, \ldots, s$ in an increasing manner, and consider the eligible items that end at the segment F_j one-by-one. When considering an item e, starting at F_i and ending at F_j, we set $M(i, j) = 1$ if $i < j$, and for each pair (i', j') where $1 \leq i' < i \leq j' < j$ and $M(i', j') \neq \star$, we set $M(i', j) = M(i', j') + 1$, unless this would increase the value of $M(i', j)$.

Next, for each $i = 0, \ldots, s$, we compute a smallest possible set $T(i)$ such that the family $\mathcal{F}$ of equality segments for $F \cup T(i)$ are such that (i) $F_{i+1}, \ldots, F_s \in \mathcal{F}$, and (ii) all segments in $\mathcal{F} \setminus \{F_{i+1}, \ldots, F_s\}$ are even; if no such set exists, we write $T(i) = \star$. We set $T(0) = \emptyset$. For some i, if $|F_i|$ is even, then we clearly have $T(i) = T(i-1)$. By contrast, if $|F_i|$ is odd, then we let $T(i)$ be a smallest possible set of the form $T(j-1) \cup M(j, i)$ where $1 \leq j \leq i$ (we require $M(j, i) \neq \star$ and $T(j-1) \neq \star$ as well in order to get a well-formed set $T(i)$; if this is not possible, we set $T(i) = \star$). The correctness of this formula is straightforward using the definitions of the sets $T(i)$ and $M(j, i)$. Finally, observe that $T(s)$ is the solution we are aiming for, so the algorithm outputs $T(s)$ if it is a subset of E, and 'No' otherwise. The running time is clearly polynomial in the input size. □

4.2 Three Agents

The complexity of checking if an envy-free allocation exists for a constant number of agents has been an open problem [Bouveret *et al.*, 2010; Aziz *et al.*, 2015b].We determine the complexity of the problem as polynomial-time solvable for two agents, but NP-complete for three agents. We will use the following reformulation of envy-freeness.

Proposition 2. *For a given set N of agents, a set I of items, and a preference list L^A for each agent $A \in N$, an allocation $\pi : I \to N$ is envy-free if and only if for each pair of agents A and B ($A \neq B$) and index i, $1 \leq i \leq |I|$:*

$$|[L^A(1:i)] \cap \pi^{-1}(A)| \geq |[L^A(1:i)] \cap \pi^{-1}(B)|. \quad (1)$$

Theorem 4. *Deciding whether a complete envy-free allocation exists is NP-complete for an instance with 3 agents.*

Proof. Containment in NP is trivial, and we will show NP-hardness of our problem by a reduction from the NP-complete NOT-ALL-EQUAL 3SAT problem [Schaefer, 1978]. The input for NOT-ALL-EQUAL 3SAT is a CNF Boolean formula $\varphi = c_1 \wedge \cdots \wedge c_m$ with variables $x_1, \ldots, x_n$, where each clause contains three literals. The task is to find a truth assignment for φ such that each clause contains at least one true literal and at least one false literal; such an assignment is *valid*. We construct an instance (N, I, L) of ENVY-FREENESS with $N = \{A, B, C\}$ such that (N, I, L) admits a complete envy-free allocation if and only if φ has a valid assignment.

W.l.o.g. we may assume that each variable occurs an even number of times in φ (this property can be achieved by adding the clause $(x_i \vee x_i \vee \overline{x_i})$ for each variable x_i with an odd number of occurrences). We transform φ into a formula φ' as follows: for each clause $c_i = (\ell_u \vee \ell_v \vee \ell_z)$ we add the clause $c'_i = (\overline{\ell_u} \vee \overline{\ell_v} \vee \overline{\ell_z})$. Clearly, any truth assignment is valid for φ if and only if it is valid for φ'. Moreover, each variable x_i has the same (even) number, say μ_i, of occurrences as a positive and as a negative literal in φ'; note $\sum_{i=1}^{n} \mu_i = 3m$.

We are going to define the preferences of the agents through several types of "building blocks". To this end, we define a *block* as a triple of lists where each list is a linearly ordered subset of I. The *concatenation* of two blocks $L = (L_1, L_2, L_3)$ and $L' = (L'_1, L'_2, L'_3)$ is the block $L + L' = (L_1 + L'_1, L_2 + L'_2, L_3 + L'_3)$, where $L_i + L'_i$ denotes the (standard) concatenation of lists.

We begin with a single *initial block* I_0. Then, for each variable x_i, $1 \leq i \leq n$, we define the following blocks. For each occurrence of x_i, we construct a *literal block*: for some j, $1 \leq j \leq \mu_i$, we denote the literal block corresponding to the j-th occurrence of variable x_i by $X_{i,j}$. Then we construct $\mu_i/2$ *equivalence blocks* $E_{i,2j}$, $1 \leq j \leq \mu_i/2$. We denote the concatenation $X_{i,1} + \cdots + X_{i,\mu_i} + E_{i,2} + \cdots + E_{i,\mu_i}$ by Y_i. Intuitively, each literal block represents the choice of a truth assignment for the given occurrence of a variable, while the equivalence blocks will ensure that these choices are consistent. Thus, the blocks in Y_i represent the choice of a truth assignment for the variable x_i. Next, for each clause c_i of φ, we define a *validity block* V_i; this block will make sure that any complete envy-free allocation corresponds to a truth assignment that is valid for the clauses c_i and c'_i. Finally, we define a *closing block* Z. The full preference lists of the agents are then obtained by the concatenation $I_0 + Y_1 + \cdots + Y_n + V_1 + \cdots + V_m + Z$.

We give the definitions of the building blocks below. For better readability, we give each block as subsequences of the preference lists of the agents in $N = \{A, B, C\}$. Moreover, we define a *triad* as a group of three items contained in $L^X[3k+2 : 3k+4]$ for some $k \in \mathbb{Z}$ and $X \in N$. In the arguments below, it will be crucial to view the list contained in some block (other than I_0 and Z) as sequences of triads.

Block I_0:
- A: $a^3_{1,0}$
- B: $b^3_{1,0}$
- C: $c^3_{1,0}$

Block $X_{i,j}$:

A: $b^3_{i,j-1}, c^3_{i,j-1}, a^1_{i,j},\ c^1_{i,j}, \beta_{i,j}, a^2_{i,j},\ b^2_{i,j}, \gamma_{i,j}, a^3_{i,j},\ b^1_{i,j}, [ca]^1_{i,j}, [ca]^2_{i,j},\ c^2_{i,j}, [ab]^1_{i,j}, [ab]^2_{i,j},\ [bc]^2_{i,j}, [ab]^0_{i,j}, [ca]^0_{i,j}$

B: $a^3_{i,j-1}, [bc]^1_{i,j}, [bc]^2_{i,j},\ a^1_{i,j}, c^1_{i,j}, b^2_{i,j},\ a^2_{i,j}, \gamma_{i,j}, b^3_{i,j},\ c^3_{i,j-1}, \alpha_{i,j}, b^1_{i,j},\ c^2_{i,j}, [ab]^1_{i,j}, [ab]^2_{i,j},\ [ca]^2_{i,j}, [ab]^0_{i,j}, [bc]^0_{i,j}$

C: $a^3_{i,j-1}, [bc]^1_{i,j}, [bc]^2_{i,j},\ b^1_{i,j}, [ca]^1_{i,j}, [ca]^2_{i,j},\ a^2_{i,j}, b^2_{i,j}, c^3_{i,j},\ b^3_{i,j-1}, \alpha_{i,j}, c^1_{i,j},\ a^1_{i,j}, \beta_{i,j}, c^2_{i,j},\ [ab]^2_{i,j}, [ca]^0_{i,j}, [bc]^0_{i,j}$

To "attach" the blocks of some variable x_i to the blocks of the previous variable x_{i-1}, we let $a^3_{i,0} = a^3_{i-1,\mu_{i-1}}$, $b^3_{i,0} = b^3_{i-1,\mu_{i-1}}$, and $c^3_{i,0} = c^3_{i-1,\mu_{i-1}}$ whenever $i \geq 2$; we only have duplicate names for these items to ease the formalization. For similar reasons, we let $[ab]^0_{i,\mu_i+1} = [ab]^0_{i,1}$, $[ab]^1_{i,\mu_i+1} = [ab]^1_{i,1}$, and $\gamma_{i,\mu_i+1} = \gamma_{i,1}$ in the definition of $E_{i,2j}$ below (indices are taken modulo μ_i for these items).

Block $E_{i,2j}$:

A: –

B: $[ca]^1_{i,2j-1}, [ca]^0_{i,2j}, \beta_{i,2j-1},\ [ca]^0_{i,2j-1}, [ca]^1_{i,2j}, \beta_{i,2j}$

C: $[ab]^1_{i,2j}, [ab]^0_{i,2j+1}, \gamma_{i,2j},\ [ab]^0_{i,2j}, [ab]^1_{i,2j+1}, \gamma_{i,2j+1}$

For defining the validity block V_i, let us assume that clause c_i contains the j_u-th, j_v-th, and j_z-th occurrence of the variables x_u, x_v, x_z, respectively, in the formula φ. If x_u appears in c_i as a positive literal, then we define the object ℓ_u as $\ell_u = [bc]^1_{u,j_u}$, otherwise we set $\ell_u = [bc]^0_{u,j_u}$. We define ℓ_v and ℓ_z analogously, and we denote the items corresponding to the negated form of these literals by $\overline{\ell_u}$, $\overline{\ell_v}$, and $\overline{\ell_z}$ (thus, if $\ell_u = [bc]^1_{u,j_u}$, then $\overline{\ell_u} = [bc]^0_{u,j_u}$, and vice versa). Now we are ready to describe the validity block V_i.

Block V_i:
- A: $s_i, \ell_u, \ell_v,\ \ell_z, t^1_i, \alpha_{u,j_u},\ \alpha_{v,j_v}, \overline{\ell_u}, \overline{\ell_v},\ \overline{\ell_z}, t^2_i, \alpha_{z,j_z}$
- B: s_i, t^1_i, t^2_i
- C: s_i, t^1_i, t^2_i

Block Z:
- A: $b^3_{n,\mu_n}, c^3_{n,\mu_n}$
- B: $a^3_{n,\mu_n}, c^3_{n,\mu_n}$
- C: $a^3_{n,\mu_n}, b^3_{n,\mu_n}$

It is straightforward that the construction takes polynomial time; note that $|I| = 66m + 3$. To verify its correctness, let us first suppose that $\pi : I \to N$ is a complete envy-free allocation. We need the following statements.

Lemma 1. *Suppose π is a complete envy-free allocation for (N, I, L).*

(i) For all indices i and j, $1 \le i \le n, 1 \le j \le \mu_i$, for each $h \in \{1,2,3\}$, and for each k with $1 \le k \le m$, we have

$$\begin{aligned} &\pi(a^h_{i,j}) = A, \quad \pi(\alpha_{i,j}) = A, \quad \pi(s_k) = A. \\ &\pi(b^h_{i,j}) = B, \quad \pi(\beta_{i,j}) = B, \\ &\pi(c^h_{i,j}) = C, \quad \pi(\gamma_{i,j}) = C, \end{aligned}$$

(ii) In any literal block $X_{i,j}$, one of the followings hold:
(C1) $X_{i,j}$ is of type 1*, meaning*

$$\begin{aligned} &\pi([bc]^1_{i,j}) = C, \quad \pi([bc]^2_{i,j}) = B, \quad \pi([bc]^0_{i,j}) = B, \\ &\pi([ca]^1_{i,j}) = A, \quad \pi([ca]^2_{i,j}) = C, \quad \pi([ca]^0_{i,j}) = C, \\ &\pi([ab]^1_{i,j}) = B, \quad \pi([ab]^2_{i,j}) = A, \quad \pi([ab]^0_{i,j}) = A, \end{aligned}$$

(C2) $X_{i,j}$ is of type 2*, meaning*

$$\begin{aligned} &\pi([bc]^1_{i,j}) = B, \quad \pi([bc]^2_{i,j}) = C, \quad \pi([bc]^0_{i,j}) = C, \\ &\pi([ca]^1_{i,j}) = C, \quad \pi([ca]^2_{i,j}) = A, \quad \pi([ca]^0_{i,j}) = A, \\ &\pi([ab]^1_{i,j}) = A, \quad \pi([ab]^2_{i,j}) = B, \quad \pi([ab]^0_{i,j}) = B. \end{aligned}$$

(iii) For any i, $1 \le i \le n$, all literal blocks in Y_i are of the same type; we call this the type of Y_i.

(iv) Let S be the list $L^X[1 : 3k{+}1]$ for some $k \in \mathbb{N}$ and agent X, where either $X = A$ and $k \le 18m$, or $X \in \{B, C\}$ and $k \le 21m$. Then $[S]$ contains exactly $k + 1$ items allocated to X by π, and exactly k items allocated to each of the other two agents.

(v) Let S be the list $L^A[1 : 54m + 6k + 1]$ for some $k \in \{1, \dots, 2m\}$. Then $[S]$ contains exactly $18m + 2k + 1$ items allocated to A by π, and exactly $18m + 2k$ items allocated to each of the agents B and C.

Proof. We prove the lemma in an inductive manner, block by block. Within a block, however, we will move from triad to triad. Let us consider such prefixes S_A, S_B, and S_C of the preference lists L^A, L^B, and L^C, respectively, for which $\mathcal{B} = (S_A, S_B, S_C)$ is the concatenation of the first few blocks in our constructed instance, and let B_{next} be the next block. We prove the lemma by induction, so we assume that the statements of (i) and (ii) hold for all items appearing in $\mathcal{B}$, and that (iv) and (v) hold for all lists S contained in $\mathcal{B}$.[1] We refer to these claims as the *induction statements*, to distinguish them from the statements of the lemma.

First observe that the induction statements indeed hold if $\mathcal{B} = I_0$. To see this, observe that in an envy-free complete allocation each agent must get its most preferred item.

Now, we are going to prove that the induction statements also hold for $\mathcal{B} + B_{\text{next}}$. We distinguish between the following cases, depending on B_{next}.

Case for a literal block: $B_{\text{next}} = X_{i,j}$ for some i and j.

By the induction, we know $\pi(a^3_{i,j-1}) = A$, $\pi(b^3_{i,j-1}) = B$ and $\pi(c^3_{i,j-1}) = C$. Also, (iv) holds for S_A, S_B, and S_C, so each of the agents has to obtain at least one item from his or her three most preferred items in $X_{i,j}$ to ensure envy-freeness. Therefore, the first triad for A shows that π must allocate $a^1_{i,j}$ to A. Also, one of $[bc]^1_{i,j}$ and $[bc]^2_{i,j}$ must be allocated to B, and the other to C. Then, looking at the second triads for B and C in $X_{i,j}$, we get that $\alpha_{i,j}$ can only be allocated to A, so as not to create too many items in the preference list of B allocated to C, or vice versa. This yields also $\pi(b^1_{i,j}) = B$ and $\pi(c^1_{i,j}) = C$. Now, considering agents A and C and their second and third triads in $X_{i,j}$, resp., we get that one of $[ca]^1_{i,j}$ and $[ca]^2_{i,j}$ must be allocated to A, and the other to C. Considering the third triad for agent B, $\pi(b^2_{i,j}) = B$ follows. Next, looking at the third triad for A and the fourth triad for C, we can observe that $\beta_{i,j}$ must be allocated to B to ensure envy-freeness, and $\pi(a^2_{i,j}) = A$ and $\pi(c^2_{i,j}) = C$ follow as well. By the fourth triads for A and B, one of $[ab]^1_{i,j}$ and $[ab]^2_{i,j}$ must be allocated to A, and the other to B. Considering the fifth triads, arguing as above we get $\pi(a^3_{i,j}) = A$, $\pi(b^3_{i,j}) = B$ and $\pi(c^3_{i,j}) = \pi(\gamma_{i,j}) = C$. This shows the induction statement for (i).

Now, consider the last triads of $X_{i,j}$. Clearly, each agent has to be allocated at least one item from his or her triad, and there are exactly three items ($[bc]^0_{i,j}$, $[ca]^0_{i,j}$, and $[ab]^0_{i,j}$) that they can get. Supposing that π allocates both $[bc]^2_{i,j}$ and $[ca]^2_{i,j}$ to C, one can see that neither $[bc]^0_{i,j}$, nor $[ca]^0_{i,j}$ can be allocated to C, as that would create too many items allocated by π to C in the list of either A or B. Analogously, we obtain that neither $\pi([bc]^2_{i,j}) = \pi([ab]^2_{i,j}) = B$, nor $\pi([ca]^2_{i,j}) = \pi([ab]^2_{i,j}) = A$ is possible. Hence, we must have that either $\pi([bc]^2_{i,j}) = C$, $\pi([ca]^2_{i,j}) = A$ and $\pi([ab]^2_{i,j}) = B$, or $\pi([bc]^2_{i,j}) = B$, $\pi([ca]^2_{i,j}) = C$ and $\pi([ab]^2_{i,j}) = A$. In the former case, we quickly get that A cannot have $[ab]^0_{i,j}$ (as otherwise B would have two items in his last triad of $X_{i,j}$ allocated to A), yielding $\pi([ab]^0_{i,j}) = B$. Similarly, we get $\pi([bc]^0_{i,j}) = C$ and $\pi([ca]^0_{i,j}) = A$ as well. In the latter case, the analogous arguments prove $\pi([bc]^0_{i,j}) = B$, $\pi([ca]^0_{i,j}) = C$ and $\pi([ab]^0_{i,j}) = A$. Thus, we get that the induction statement for (ii) holds as well.

It remains to observe that π allocates exactly one item to each of the agents from every triad. Therefore, all the induction statements hold for $\mathcal{B} + B_{\text{next}}$.

Case for an equivalence block: $B_{\text{next}} = E_{i,2j}$ for some i and j.

Since the induction statements for claims (ii) and (iv) hold for S_B, and both $[ca]^1_{i,2j-1}$ and $[ca]^0_{i,2j}$ appear in the first triad for B, we obtain that either $\pi([ca]^1_{i,2j-1}) = A$ and $\pi([ca]^0_{i,2j}) = C$, or vice versa. Hence, $X_{i,2j-1}$ and $X_{i,2j}$ must have the same types, which shows also that each agent obtains exactly one item from both triads for B (using also that we have $\pi(\beta_{i,2j}) = \pi(\beta_{i,2j-1}) = B$ by induction). Similarly, the triads for C show that $X_{i,2j}$ and $X_{i,2j+1}$ have the same type, and that π allocates an item from each triad for C to each agent. This proves that claim (iv), and therefore all the induction statements as well, hold for $\mathcal{B} + B_{\text{next}}$.

Case for a validity block: $B_{\text{next}} = V_i$ for some i.

By the induction statement for claim (ii), we know that each of the items ℓ_u, ℓ_v, and ℓ_z is allocated to one of B

[1] More precisely, we assume that (iv) and (v) hold for all lists S that are of the form specified by the corresponding statement, and which, additionally, are contained in one of S_A, S_B, or S_C. Note that the statement of (v) is empty if $|S_A| \le 54m + 1$, meaning that $\mathcal{B}$ does not contain any validity blocks.

or C by π. Thus, at least two of these items must be allocated to the same agent, and since (iv) and (v) hold for S_A, we obtain $\pi(s_i) = A$. Thus, from the triads for B and C, we get that π allocates one of t_i^1 and t_i^2 to B, and the other to C. By the induction statement for claim (i) we know $\pi(\alpha_{u,j_u}) = \pi(\alpha_{v,j_v}) = \pi(\alpha_{z,j_z}) = A$, from which it follows that π allocates exactly two items to each of the agents from the first two triads for A. Similarly, each agent gets two items from the last two triads for A. This proves the induction statements for this case.

Case for the closing block: $B_{\text{next}} = Z$.

Notice that we only need to prove the induction statements for claims (iv) and (v) here, which hold trivially. Using the induction statements for the whole instance we obtain claims (i), (ii), (iv), and (v) immediately. Finally, our arguments for the case of an equivalence block also prove claim (iii). □

Using Lemma 1 we can construct a valid truth assignment for φ' based on the allocation π. Namely, we set x_i to true if and only if the literal blocks in Y_i are of type 1; by Claim (iii) of Lemma 1 π is well-defined.

Consider the validity block V_i for some $1 \leq i \leq m$, involving the j_u-th, j_v-th, and j_z-th occurrence of the variables x_u, x_v, and x_z, respectively. Note that there are exactly $54m + 12(i-1) + 1$ items preceding block V_i in the preference list L^A of agent A. By Claim (v) of Lemma 1, we know that among these items exactly $18m + 4(i-1) + 1$ are allocated to A by π, and exactly $18m + 4(i-1)$ are allocated to each of the other two agents. By Claim (i) of Lemma 1, we also know $\pi(s_i) = \pi(\alpha_{u,j_u}) = \pi(\alpha_{v,j_v}) = \pi(\alpha_{z,j_z}) = A$, and from Claim (ii) we also get that each of ℓ_u, ℓ_v, and ℓ_z is allocated to one of the agents B or C. By Claim (v), we know that π allocates exactly two of the items ℓ_u, ℓ_v, ℓ_z, and t_i^1 to B, leaving the other two items for C. Similarly, the same holds for the items $\overline{\ell_u}$, $\overline{\ell_v}$, $\overline{\ell_z}$, and t_i^2.

Using now the definition of these items, and that condition (C1) holds for the variables set to true, we get that the number of true literals in the clause c_i equals the number of items in $\{\ell_u, \ell_v, \ell_z\}$ allocated to C by π. Since this value must be either 1 or 2 (as argued above), we get that c_i contains at least 1 but at most 2 true literals. Similarly, we obtain the same for c_i', which proves that our truth assignment is indeed valid for φ', and hence for φ.

For the converse direction, suppose that we are given a valid truth assignment ρ for φ. We construct an allocation π as follows. First, we allocate all items appearing in Claim (i) of Lemma 1 as required there. Next, for each variable x_i, we let Y_i have type 1 exactly if ρ sets x_i to true, and we let Y_i have type 2 otherwise (yielding the allocations as given in Claim (ii) of Lemma 1). Finally, we set $\pi(t_i^1) = B$ and $\pi(t_i^2) = C$ if there are 2 true literals in the clause c_i according to ρ, and we set $\pi(t_i^1) = C$ and $\pi(t_i^2) = B$ otherwise. It is straightforward to verify the envy-freeness of π, using the characterization given in Prop. 2. □

Corollary 1. *With three agents, adding/deleting/replacing the minimum number of items to ensure the existence of an envy-free allocation is NP-hard.*

5 Proportionality

Assuming cardinal utilities, an assignment is *proportional* if each agent gets at least $1/|N|$-th of the utility of all the items. In ordinal settings, we will say that an assignment is *necessarily proportional* if it is proportional for all cardinal utilities consistent with the ordinal preferences. Aziz *et al.* [2015b] showed that itemwise envy-freeness implies necessary proportionality, and that for two agents, necessary proportionality is equivalent to (itemwise) envy-freeness. Hence, all the results for envy-freeness for two agents carry over for necessary proportionality for two agents.

Corollary 2. *With two agents, the problem of adding/deleting the minimum number of items to ensure the existence of a necessarily proportional allocation is polynomial-time solvable.*

With more than two agents, there exists a polynomial-time algorithm to check whether there exists a necessarily proportional assignment. This raises the question whether the problem of adding/deleting items to achieve necessary proportionality is also polynomial-time solvable.

6 Envy and utility

A weaker form of envy-freeness considers the utilities that agents have for their items. We say that an allocation is *envy-free w.r.t. utilities* if for each agent A, the sum of utilities that A has for its items is at least as great as the sum of utilities that A has for the items allocated to any other agent. Supposing additive utilities, itemwise envy-freeness ensures envy-freeness w.r.t. any utilities consistent with the preference orderings of the agents over the items. As envy-freeness w.r.t. utilities is a weaker property than itemwise envy-freeness, it may be easier to achieve and may require adding or deleting fewer items.

There are, however, several disadvantages to considering envy-freeness w.r.t. utilities compared to itemwise envy-freeness. First, the chair would need to elicit utilities from the agents. Even supposing additive utilities, this is more challenging than eliciting just a preference ordering over items. Second, even with just two agents and identical utilities for the agents, deciding if there is an envy-free allocation w.r.t. utilities is NP-hard (by a simple reduction from integer partitioning). Therefore, adding or deleting a minimum number of items to ensure envy-freeness w.r.t. utilities is NP-hard even with only two agents. This contrasts our results that, with two agents, it takes polynomial time to add or delete a minimum number of items to ensure itemwise envy-freeness.

7 Discussion

In this paper, we have proposed a new research direction: the algorithmic and computational aspects of control in fair allocation. As a case study, we presented algorithmic results for ensuring envy-freeness by adding or deleting items for the case of two agents. We also settled an open problem, by proving that checking whether there exists an envy-free allocation is NP-complete for the case of three agents. Our discussion raises a number of interesting research questions for other control problems with other possible goals.

Acknowledgments

Data61 (formerly known as NICTA) is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program. Ildikó Schlotter is supported by the Hungarian Scientific Research Fund (OTKA grants no. K-108383 and no. K-108947). Toby Walsh is now supported by the European Research Council via AMPLIFY 670077.

References

[Aziz *et al.*, 2014] H. Aziz, S. Gaspers, S. Mackenzie, and T. Walsh. Fair assignment of indivisible objects under ordinal preferences. In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1305–1312, 2014.

[Aleksandrov *et al.*, 2015a] M. Aleksandrov, H. Aziz, S. Gaspers, and T. Walsh. Online fair division: Analysing a food bank problem. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2540–2546, 2015.

[Aziz *et al.*, 2015b] H. Aziz, S. Gaspers, S. Mackenzie, and T. Walsh. Fair assignment of indivisible objects under ordinal preferences. *Artificial Intelligence*, 227:71–92, 2015.

[Bartholdi, III *et al.*, 1992] J. Bartholdi, III, C. A. Tovey, and M. A. Trick. How hard is it to control an election? *Mathematical and Computer Modelling*, 16(8–9):27–40, 1992.

[Bouveret *et al.*, 2010] S. Bouveret, U. Endriss, and J. Lang. Fair division under ordinal preferences: Computing envy-free allocations of indivisible goods. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI)*, pages 387–392, 2010.

[Bouveret *et al.*, 2016] S. Bouveret, Y. Chevaleyre, and N. Maudet. Fair allocation of indivisible goods. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia, editors, *Handbook of Computational Social Choice*, chapter 12. Cambridge University Press, 2016.

[Brams and Taylor, 1996] S. J. Brams and A. D. Taylor. *Fair Division: From Cake-Cutting to Dispute Resolution*. Cambridge University Press, 1996.

[Brams *et al.*, 2014] S. J. Brams, D. M. Kilgour, and C. Klamler. Two-person fair division of indivisible items: An efficient, envy-free algorithm. *Notices of the AMS*, 61(2):130–141, 2014.

[Cechlárová and Schlotter, 2010] K. Cechlárová and I. Schlotter. Computing the deficiency of housing markets with duplicate houses. In *Proceeding of the 5th International Symposium Parameterized and Exact Computation (IPEC)*, pages 72–82, 2010.

[Moulin, 2003] H. Moulin. *Fair Division and Collective Welfare*. The MIT Press, 2003.

[Nguyen and Vohra, 2014] T. Nguyen and R. Vohra. Near feasible stable matchings with complementarities. Technical report, SSRN, 2014.

[Procaccia and Tennenholtz, 2013] A. D. Procaccia and M. Tennenholtz. Approximate mechanism design without money. *ACM Transactions on Economics and Computation*, 1(4), Article no. 18, 2013.

[Procaccia and Wang, 2014] A. D. Procaccia and J. Wang. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the 15th ACM Conference on Economics and Computation (ACM-EC)*, pages 675–692. ACM Press, 2014.

[Schaefer, 1978] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–226. ACM Press, 1978.

[Segal-Halevi *et al.*, 2015] E. Segal-Halevi, A. Hassidim, and Y. Aumann. Waste makes haste: Bounded time protocols for envy-free cake cutting with free disposal. In *Proceedings of the 14th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 901–908. IFAAMAS, 2015.