Sapienza
Università di Roma

On the polynomial computation of EFX allocations for 3 agents and 3-valued instances

Facoltà di Ingegneria dell'informazione, informatica e statistica
Corso di Laurea Magistrale in Engineering in Computer Science

Candidate
Francesco Montano
ID number 1744183

Thesis Advisor
Prof. Aris Anagnostopoulos

Co-Advisor
Dr. Georgios Birmpas

Academic Year 2020/2021

Thesis not yet defended

On the polynomial computation of EFX allocations for 3 agents and 3-valued instances
Master's thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Version: January 13, 2022

Author's email: montano.1744183@studenti.uniroma1.it

Abstract

Fair division is the problem of dividing a set of resources among a set of agents in a fair manner. In the past years several fairness criteria have been introduced, with "envy freeness" to be among the most important ones. Informally, an allocation is envy free when no agent envies the share that another agent got. The main drawback of the envy-freeness criterion, is that it cannot always be guaranteed in the setting of indivisible items. For this reason, some more relaxed criteria have been introduced by the literature, two of which are envy freeness up to one good (EF1), and a stronger version of the same concept, namely, envy freeness up to any good (EFX). Whereas there exists a polynomial time algorithm to obtain EF1 allocations for any instance, the EFX criterion remains currently more elusive, as there are only partial results regarding its existence. In this thesis, we are making steps towards improving the current state of the art for this problem, by presenting the first polynomial time algorithm for the computation of EFX allocations, under instances with three players and three values with a constraint on the values.

Contents

Chapter 1

Introduction

Fair division is the problem of dividing a set of resources between a set of players in order to satisfy some criterion of fairness. The theory of fair division dates back to the second world war when three Polish mathematicians, Hugo Steinhaus, Bronisław Knaster and Stefan Banach started working on the fair cake-cutting problem to find a generalization for any number of players of the divide and choose algorithm; the three developed the last diminisher procedure. This problem arises in several real scenarios as distributing tasks, dividing goods [Gho+18], frequency allocation, airport traffic management, and the fair and efficient exploitation of Earth Observation Satellites [BL08]. There are several application in order to deal with such problems and one of them is the spliddit site that returns provably fair solutions for rent sharing, task distributions and other problems. For instance this site for the sharing rent problem, rather than divide the rent equally among the players, he asks to each of them how much they value each room and returns the allocation of the rooms and how much each player has to pay.

Sometimes other than require some fairness criteria, is required to divide the set of resources between the players by considering also two other aspects: Pareto optimality and truthfulness [Wik21]. Pareto optimality, or Pareto efficient, is a concept introduced in Economy and game theory that occurs when the allocation of the resources is such that we cannot obtain a Pareto improvement: a player cannot improve his utility without reducing the utility of another player. Truthfulness is also a concept introduced in game theory in cases in which the players have private information, we say that an asymmetric game is truthful if disclose the private information is a weakly-dominant strategy: a strategy that provides for all the players at least the same utility that they will get by playing other strategies.

Fair division problem can be divided in several manners considering the type of items:

- divisible and indivisible items,
- homogeneous and heterogeneous items,
- goods or bads: items with positive or negative value for the players. An example of bads can be the house chores.

In this thesis I am going to approach mainly the problem of indivisible items, but I will first introduce briefly the results obtained in the field of divisible items and then I am going to describe the different type of fairness criteria.

1.1 Divisible Items

When considering the case of divisible items the main problem that we can think of is the fair cake-cutting problem. In this type of problem, we have to divide a heterogeneous resource between different agents. An example of this problem is when we have to divide a cake with several toppings between different people. In this problem, we consider that the resource can be divided into arbitrary small parts without destroying the value. The players to which we have to assign parts of the resource can have different preferences over the resource, so following the example of a cake with different toppings onto, we can say that the players can have different tastes. The cake is not the only example of this type of problem, land estates, advertisement space and broadcast time are few of other examples in which this type of problem raises.

As said in the beginning, Hugo Steinhaus, Bronisław Knaster and Stefan Banach have developed an algorithm called diminisher procedure that produces proportional division of an heterogeneous resource between n players, so that each agent receives a part of the complete resource that he values at least $\frac{1}{n}$ of the entire value of the resource. This algorithm is an extension of the cut and choose algorithm for divisible items. The cut and choose algorithm produces EF allocation for two players by letting one player divide the resource into two parts and letting the other player choose his part. We can notice that this simple algorithm produce an EF allocation since the first player will divide the resource in such a way that he is indifferent between the two parts, and the second player will take the part with higher value for him. We can also notice that such type of algorithm is not optimal, let's say that the resource is a cake with two toppings: half chocolate and half vanilla, and that the first player prefers the chocolate, while the second one prefers the vanilla topping. In this case the first player will divide the cake so that each part has half vanilla and half chocolate, while the optimal solution was to give the chocolate half to the first player and the other one to the second one [Wik21].

1.2 Indivisible Items

The subjective theory of value is a theory that proposes the idea that the value of each object is not defined by a property or by the production method, but instead has value determined by the importance that people give to it, for this reason in the fairness theory we tend to consider subjective concepts of fairness[Wik21]. Before introducing some of the most used fairness criterion let's assume that we have a set of n agents $N = \{1, 2, \ldots, n\}$ and a set M of m indivisible items. Moreover, because of the subjective concept of fairness we define with v_i the subjective utility function of player i that assigns to each possible subset $\bar{M}$ of items of M a value $v_i(\bar{M})$. In the next pages I am going to consider that the utility functions of the players are additive functions.

1.2.1 Proportionality and Envy Freeness

We say that an allocation $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ is proportional if for any player p_i holds that $v_i(A_i) \geq \frac{v_i(M)}{n}$. Another criteria, other than proportionality, is Envy freeness that has been introduced in [Fol67] [Var74] and we have that an allocation $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ is envy-free (EF) if for every couple of players $i, j \in N,\ v_i(A_i) \geq v_i(A_j)$.

We can notice that when we have indivisible items there is not always an EF allocation or a proportional one, indeed a simple counterexample for both fairness notions is a case in which we have more than one player and only one good. In this case we have that if there are at least two players that value the only item with a value strictly larger than zero than at least one of the two, the player p_i who does not get such item, will not respect the constraint of envy-freeness and proportionality because $v_i(A_i) = v_i(\emptyset) = 0$. For this reason have been introduced some relaxations of the envy-freeness and proportionality criteria: envy freeness up to one good, envy freeness up to any good, maximin share and pairwise maximin share fairness. For all of the following relaxations there exists the approximated version: we say that an allocation is α-criteria if $v_i(A_i) \geq \alpha$ *original constraint* with $\alpha \in [0, 1]$.

1.2.2 Envy Freeness up to One Good

The first introduced relaxation of the envy-free criteria has been the envy-free up to one good criteria. This relaxation has been introduced in [Lip+04] and we have that an allocation $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ is envy-free up to one good (EF1) if for every couple of players $i, j \in N$ with $A_j \neq \emptyset$, $\exists g \in A_j : v_i(A_i) \geq v_i(A_j \setminus \{g\})$.

This relaxation of the envy-free criteria has been extensively studied and today are well known two algorithms: round robin and envy cycle elimination that respectively produce EF1 allocation in the case of additive and arbitrary monotonic valuation functions.

Round Robin Algorithm Is well known an algorithm to obtain EF1 allocations with n players with additive valuation function and m items, such algorithm is the round robin algorithm. Is easy to prove that such algorithm produces an EF1 allocation: let's consider

Algorithm 1: Round Robin Algorithm

```
S ← M;
while S ≠ ∅ do
    for p_i ∈ N do
        g ∈ arg max_{x∈S} v_i(x);
        S ← S \ {g};
        A_i ← A_i ⋃{g};
```

a couple of players i, j such that j chooses before i. As first thing we have to notice that $|A_j| \geq |A_i|$ and that we can have only two cases: $|A_i| = |A_j|$ or $|A_j| = |A_i| + 1$. Now let's sort the items inside A_j and A_i with respect to the function v_i. In both cases we have that the condition of *EF1* must hold since we will have that $v_i(A_{i,k}) \geq v_i(A_{j,k+1})$ (where $A_{x,y}$ is the y-th item in the sorted set A_x) since the player i surely chooses the k-th item before the player j chooses $k + 1$ items. So in both cases is enough to use $g = A_{j,0}$ in order to have that the requirement holds. As we can see the only difference between the case in which $|A_i| = |A_j|$ and the one in which $|A_j| = |A_i| + 1$ is that in the first one the last item of the sorted set A_i, $A_{i,|A_i|}$, does not have a related $A_{j,|A_i|+1}$, while in the latter case it has. So also in both cases, by assigning $g = A_{j,0}$ (ignoring that item) we have that

$$\sum_{k=1}^{|A_j|} v_i(A_{j,k}) \leq \sum_{k=0}^{|A_i|} v_i(A_{i,k})$$

If instead i chooses before j, player i will always chose before j so he will always have $v_i(A_i) \geq v_i(A_j)$ and so the requirement to be *EF1* holds also in this case.

Envy Cycle Elimination More generally in [Lip+04] has been developed also the Envy Cycle Elimination Algorithm to compute EF1 allocations in the case in which the agents have arbitrary monotonic utilities, so in the case in which

$$v_i(S) \leq v_i(T)\ \forall S \subseteq T \subseteq M\ \forall i \in N$$

This algorithm is based on the definition of envy-graph of a partial allocation $\bar{\mathcal{A}} = (\bar{A}_1, \bar{A}_2, \dots, \bar{A}_n)$ as a graph where the nodes are the n players and there is a direct edge between p_i and p_j only if p_i envies p_j: $v_i(A_i) < v_i(A_j)$. From this definition we can notice that if p_i is a source of the graph, he is not envied by any player, so if we start from a partial EF1 allocation, construct its envy-graph and assign one not allocated item to a player that is a source, the result is still an EF1 allocation since no agent envies the bundle of this player when we remove the last added item because it was a source. It can happen that there is no source in the envy-graph, than if there are edges, there must be a cycle: in order to deal with the cycles is enough to rotate the bundles along the cycle. When we remove a cycle we can notice that, since no other edge will be created because we are not changing the bundles but only changing the owner and each player involved in the bundle swap will increase its utility, the number of edges in the graph will decrease at least by the number of edges in the cycle and this implies the fact that the algorithm terminates.

Algorithm 2: Envy-Cycle Elimination Algorithm

```
S ← M;
Ā ← (∅, ∅, ..., ∅);
while S ≠ ∅ do
    while There is a cycle do
        remove such cycle;
    //since there are no cycles there must be a source;
    allocate item g ∈ R to a source player;
```

1.2.3 Maxmin Share

This fairness criteria has been introduced in [Bud10] and is a relaxation of the proportionality criteria introduced for the same reason of the EF1 criteria: with indivisible items we cannot always reach proportionality. In order to describe this cirteria we have to introduce the definition of *n-maximin share* of agent i as $\boldsymbol{\mu}_i = \max_{\mathcal{A} \in \prod_n(M)} \min_{A_j \in \mathcal{A}} v_i(A_j)$. We say that an allocation $\mathcal{A} = (A_1, A_2, \dots, A_n)$ respect the maxmin share criterion if

$$\forall i \in N:\ v_i(A_i) \geq \boldsymbol{\mu}_i$$

1.2.4 Pairwise Maximin Share Fairness

Introduced in [Car+19], we say that an allocation $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ respects the pairwise maximin share (PMMS) criteria if, considering $(B_i, B_j) \in \prod_2(A_i \cup A_j), \forall i, j \in N$ holds that

$$v_i(A_i) \geq \max_{(B_1,B_2)} \min\{v_i(B_1), v_i(B_2)\}$$

1.2.5 Envy Freeness up to Any Good

Envy Freeness up to any good is a relaxation of the envy-free criteria that is much stronger than the EF1 and has been introduced because this kind of allocation are good approximation to the EF ones. This relaxation has been introduced in [Car+19] [GMT14] and we have that an allocation $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ is envy-free up to any good (EFX) if for every couple of players $i, j \in N$ with $A_j \neq \emptyset$, $\forall g \in A_j : v_i(A_i) \geq v_i(A_j \setminus \{g\})$.

In the last years this criteria has been extensively studied:

- in 2016 has been given a formal definition,
- in 2018 has been shown that with the divide and choose algorithm [PR20a] we can obtain EFX for two players or n players with identical valuation functions.
- in 2019 has been shown that we can build allocations that are EFX and have at least half of the maximum possible Nash Welfare by assigning to the agents only a subset of the items and giving the remaining ones to charity [CGH19].
- in 2020 has been shown that for 3 players always exists an EFX allocation [CGM20].

So summarizing till now we have the following results with respect to the number of agents:

- 2 players: the divide and choose algorithm [PR20a] produces an EFX allocation in polynomial time.
- 3 players: always exists an EFX allocation, but till now we only have a pseudo-polynomial algorithm [CGM20].
- ≥ 4 players: there exists an EFX allocation if we consider only a subset of the entire set of items [CGH19].

Other than this, in [BK20] has been shown that when the values of the items have the same ordering for the players than EFX allocation exists and that such EFX allocation is also a $\frac{2}{3}$-MMS allocation. Instead in [PR20b] has been proposed an algorithm to obtain $\frac{1}{2}$-EFX allocation when players have sub-additive valuations. Finally in [Ama+21] has been proposed a modified version of the round robin algorithm that obtains EFX allocation in the case of additive valuation functions where for each player p_i the values of the items in M have value in the range $[x_i, 2x_i]$ $x_i \in R_{>0}$.

Cut and Choose Algorithm The cut and choose algorithm introduced, for indivisible items, in [PR20a] produces EFX allocation for two players with additive valuation functions and is described in 3. In this algorithm we use the *Leximin++Solution* function that, given the number of partitions, the set of items, and the valuation function of p_1 returns, in the case of 2 players, two set of items that are EFX for p_1.

Algorithm 3: Cut and Choose Algorithm

$(A_1, A_2) \leftarrow$ Leximin++Solution$(2, M, v_1)$;
if $v_2(A_1) \geq v_2(A_2)$ **then**
    return (A_2, A_1);
else
    return (A_1, A_2);

1.2.6 Relations among the fairness criteria

$$
\begin{tikzcd}
 & & & \frac{2}{3}\text{-}PMMS \arrow[r, Rightarrow] & \frac{1}{2}\text{-}EF1 \arrow[r, Rightarrow] & \frac{1}{2n-1}\text{-}MMS \\
EF \arrow[r, Rightarrow] & PMMS \arrow[r, Rightarrow] & EFX \arrow[r, Rightarrow] \arrow[ur, Rightarrow] \arrow[dr, Rightarrow] & \frac{4}{7}\text{-}MMS & \frac{1}{n}\text{-}MMS & \frac{1}{3}\text{-}EF1 \\
 & & & EF1 \arrow[r, Rightarrow] \arrow[ur, Rightarrow] & \frac{1}{2}\text{-}PMMS \arrow[r, Rightarrow] \arrow[ur, Rightarrow] & \frac{1}{3n-2}\text{-}MMS
\end{tikzcd}
$$

Figure 1.1. Relations among the fairness criteria

The relations shown in Figure 1.1 is the summary of several works: in [Car+19] has been shown that $EF \implies PMMS \implies EFX \implies EF1$, instead in [ABM18] has been shown that in the case of $n \in \{2, 3\}$ $EFX \implies \frac{2}{3}$*-MMS*, instead in the case of $n \geq 4$ $EFX \implies \frac{4}{7}$*-MMS*.

In [ABM18] has also been analyzed which kind of allocation are implied when we have an α-EF1 or α-PMMS allocation. In the case of α-EF1 allocation they proved that for any $\alpha \in (0, 1]$ an α-EF1 allocation is also a $\frac{\alpha}{n-1+\alpha}$-MMS allocation and that α-EF1 $\implies$ $\frac{\alpha}{1+alpha}$-PMMS for $n \geq 3$. In the case of α-PMMS allocation they proved that for any $\alpha \in (0, 1)$ we have that $\alpha\text{-}PMMS \implies \frac{\alpha}{2-\alpha}\text{-}EF1$ and that considering also $n \geq 3$ any α-PMMS is also an $\frac{\alpha}{2(n-1)-\alpha(n-2)}$-MMS.

1.2.7 Thesis Objective

In this thesis I am going to first show that, starting from the Match&Freeze algorithm, we can obtain EFX allocation for two players and valuation functions that value each item with one out of three possible values. After this, since we already have an algorithm that, given two players and more general valuation functions, produces EFX allocation, I am going to show how to obtain EFX allocation in polynomial time for three players and valuation functions that value each item with one out of three possible values with some other constraint.

Chapter 2

Three Values and Two Players

The Match and Freeze algorithm 4 is an algorithm introduced in [Ama+21] that computes EFX allocations for n players with additive valuation functions with two values a, b such that $a > b \geq 0$. This algorithm proceeds in rounds and keeps two sets: M and L respectively the set of unallocated items and the set of active players. In each round, we assign to each active player one item from the M set, except in the last round in which some player could not receive any item. At each round we decide the item to assign to each player by computing the maximum matching over the bipartite graph where the nodes are the active players in L and the unallocated goods in M and the edges are such that $(i, g) \in E$ if $i \in L, g \in M, v_i(g) = a$. Can happen that some agent are not assigned to any item by the maximum matching, in such a case we assign him a arbitrary available good. This can happen to player i for two reasons:

- there are no items in M valued a by p_i
- there are items in M valued a by p_i, but the maximum matching assigned them to other players.

Only the second case is relevant in order to obtain EFX allocations: agent i could value lower its bundle than the one of the players who took the items that he valued a in the last iteration. In order to solve this we freeze for $\left\lfloor \frac{a-b}{b} \right\rfloor$ rounds the players that took the item valued a by p_i so that he can regain the lost value. In such algorithm we not only freeze the agents that took an item valued a by a player i while p_i took an item valued b, we also freeze all the agents that took an item valued a by one of players that are going to be frozen at the end of the round. So we define the frozen set F as $F = \{i \in L | \exists j \in L : v_j(g_i) = a, v_j(g_j) = b\}$ where g_k is the item took in the current round by player p_k.

2.1 Counter Example for Three Values for The Match&Freeze Algorithm

We can see in the next example that the algorithm does not work with three values valuation functions even with only two players. The match and freeze counter example for three values is the following one: let's consider the case in which there are $m = 5$ items and two players p_1, p_2 with the valuations of the items expressed in Table 2.1. By

Algorithm 4: Match&Freeze Algorithm

$L \leftarrow N$;
$R \leftarrow M$;
$l = (1, 2, \ldots, n)$ **while** $R \neq \emptyset$ **do**
    Construct the bipartite graph $G = (L \cup R, E)$;
    Compute the maximum matching on G;
    for *each matched pair* (i, g) **do**
        Allocate good g to agent i;
        Remove g from R;
    for *each unmatched active agent* i *w.r.t.* l **do**
        Allocate one arbitrary unallocated good g to i;
        Remove g from R;
    Construct the set F of agents that need to freeze;
    Remove agents of F from L for the next $\lfloor \frac{a}{b} \rfloor - 1$ rounds;
    Put agents of F to the end of l;
return the resulting allocation $\boldsymbol{A}$;

considering $a = 100$, $b = 50$ and $c = 1$ the output of the algorithm is $A_1 = \{i_1\}$, $A_2 = \{i_2, i_3, i_4, i_5\}$ where A_i is the sets of items p_i takes. This is not an EFX allocation since $v_1(A_1) = 100 < v_1(A_2 \setminus \{i_5\}) = 150$.

	i_1	i_2	i_3	i_4	i_5
p_1	a	b	b	b	b
p_2	b	c	c	c	c

Table 2.1. Counter example for the Match&Freeze algorithm

2.2 Match&Freeze Modification

In this section we try to modify the Match and Freeze algorithm in order to obtain an EFX allocation with additive valuation functions, when there are only three possible values for the items, and we have only two players. The first thing we notice from the counter example of the previous section, is that when there are three possible values for the goods, as we freeze player p_1, it is possible for her to envy the non-frozen player p_2, since p_2 has to get $\left\lfloor \frac{b}{c} \right\rfloor - 1$ items of value c which have value b for p_1. This implies that the value of these items can be $\left\lfloor \frac{b}{c} \right\rfloor b - b$ for p_1, a value that is higher than a in case $\left\lfloor \frac{b}{c} \right\rfloor > \lfloor \frac{a}{b} \rfloor$. This would obviously lead to a non-EFX allocation in the end (from the perspective of player p_1).

2.2.1 The Modification Approach

In order to solve this problem, we will apply two modifications to the algorithm:

1. We will run the algorithm till the end, and if the produced allocation is not EFX, we will return back to the point where the first player was frozen and exchange the items between the two players (i.e. the items that were assigned to the players at this specific maximum matching)

2. We will redefine the number of iterations for which a player is frozen in order to adapt it to the three value's case. To do so, we will use a number of iterations that depends both on the remaining items, and on the values obtained by the two players. Specifically, let's consider a point where one of the players will become frozen. Without loss of generality, suppose that p_1 takes item x and p_2 took item y, the cases in which we freeze player p_1 are the following:

- $v_2(x) = a,\ v_2(y) = b$ in this case we have that p_2 can still take items with value b and c. We will unfroze p_1 only when p_2 has obtained a set of items F while p_1 is frozen such that
$$a - b \geq v_2(F) \geq a - b - c \tag{2.1}$$
The set of items F is obtained by first taking all the items with value b for p_2 by respecting that $a - b \geq v_2(F)$, then by taking items with value c for p_2 in order to respect the constraint described in equation 2.1.

  Since it is possible that there might not be enough items of value c to reach the aforementioned bound,in that case we will use a different constraint: $a - b \geq v_2(F) \geq a - 2b$. Under this new threshold two possible cases must be considered:

  - $\min_{j \in A_1} v_2(j) = b$
  - $\min_{j \in A_1} v_2(j) = c$: in this case we have that at least one item of value c for p_2 has been allocated to p_1 before she was frozen, so in the same iteration p_2 took an item of value at least b for her

- $v_2(x) = a,\ v_2(y) = c$ in this case we have that p_2 can only take items of value c, so p_1 will be frozen for $\lfloor \frac{a-c}{c} \rfloor = \lfloor \frac{a}{c} \rfloor - 1$.

- $v_l(x) = b,\ v_l(y) = c$ as in the case, we have that p_2 can only take items of value c, so p_1 will be frozen for $\left\lfloor \frac{b-c}{c} \right\rfloor = \left\lfloor \frac{b}{c} \right\rfloor - 1$.

In order to show that this approach is effective, we have to define the kind of assignment that could lead to a non-EFX allocation, and also prove that by exchanging the items of the players at the problematic point, will result to an EFX allocation.

These modifications to the original algorithm will cause that some assertions made in the original algorithm no longer will hold: the two most important are that

- a player does not always take the items following the order of the values he assign to them as in the case reported above 2.2.1

- a player could be involved in more than one freeze as we will see in 2.2.4

2.2.2 Defining the Problematic Assignments and Resolving the Cases where they do not Occur

We define a *problematic assignment* as an assignment after which a player p_i is frozen, and there are still items that p_i values as b. Now let us show that if this kind of assignment does not appear during the execution of the algorithm, the final allocation will be EFX. If we consider that there are no problematic assignments, we can have only the following two cases

1. No player will ever be frozen
2. A frozen player has a value of c for each of the remaining items

Let set $A_i[j]$ to be the j-th item took by player i. It is easy to see that in the first case we will have an EFX allocation since $v_1(A_1[l]) \geq v_1(A_2[l]) \; \forall l \in \{1, 2, \cdots, k\}$ where $k = \min(|A_1|, |A_2|)$ (the same holds for p_2). So we have that if p_1 and p_2 get the same number of item,s they are envy-free towards each other, otherwise they are EFX since the last item taken by one of the two players will have the smallest value for the other one.

In the second case we can observe that only one player will be frozen (and only once) during the execution of the algorithm. This is due to the fact that if there are no problematic assignments, when a player p_i is frozen, then then the remaining items have value c for her. This implies that she cannot be frozen again, while the maximum matching procedure guarantees that the algorithm will never freeze the other player as well. Indeed the only type of assignment that can generate a second freeze needs to have both players with an item of value b, as it can be seen in Table 2.4. We proceed by showing that in this case as well, an EFX allocation is produced.

Let's consider that at iteration i the algorithm freezes p_1, before iteration i we have that $v_1(A_1[l]) \geq v_1(A_2[l]) \; \forall l \in \{1, 2, \cdots, i-1\}$ and $v_2(A_2[l]) \geq v_2(A_1[l]) \; \forall l \in \{1, 2, \cdots, i-1\}$. Recall that this is the first time a player is frozen since, as we already mentioned, only one player can be frozen and only once during the execution of the algorithm. Let λ be the number of items that p_2 gets after p_1 is unfrozen, p_2, something that implies that p_1 will take or λ or $\lambda - 1$ items (at the end we give precedence to the player that has not been frozen). Finally, let F be the set of items that p_2 has taken while p_1 was frozen. We proceed in analyzing the different cases:

1. p_1 took item $x : v_1(x) = a$
   (a) p_2 took item $y : v_2(y) = b$ and $v_2(x) = a$
   (b) p_2 took item $y : v_2(y) = c$ and $v_2(x) = a$
   (c) p_2 took item $y : v_2(y) = c$ and $v_2(x) = b$
2. p_1 took item $x : v_1(x) = b$
   (a) p_2 took item $y : v_2(y) = c$ and $v_2(x) = b$

These are all the possible cases where p_1 can value all the remaining items as c, and as we will show, in all of these cases, the produced allocation is always EFX.[1]

[1]The symmetric cases where player p_2 is the one that is frozen, can be resolved under the same arguments.

Case 1c and 2a We begin by observing the set of items that is obtained by p_2: A_2 will be formed by A_2^* that is the set of items obtained by p_2 before that p_1 is frozen, item y obtained in the iteration in which p_1 is frozen, the set of items F that contains the items obtained by p_2 while p_1 was frozen and λ items with value c obtained from the iteration in which p_1 is unfrozen. Similarly, we have that A_1 will be formed by A_1^*, item x that p_1 obtained in the iteration that causes him to be frozen, and at most λ items with value c after she he has been unfrozen. We point out again, that it is easy to see that the latter items have value c for both players.

We let $v_2(F) > b - 2c$, so we have that,

$$\begin{aligned} v_2(A_2) &= v_2(A_2^*) + v_2(y) + v_2(F) + \lambda c \\ &= v_2(A_2^*) + c + v_2(F) + \lambda c > v_2(A_2^*) + b - c + \lambda c \end{aligned}$$

$$\begin{aligned} v_2(A_1) &= v_2(A_1^*) + v_2(x) + \lambda c \\ &= v_2(A_1^*) + b + \lambda c \leq v_2(A_2^*) + b + \lambda c \end{aligned}$$

where $v_2(A_1^*) \leq v_2(A_2^*)$ comes from the fact that this is the first freeze that happens. So the allocation produced by the algorithm in the case 1c and 2a is EFX.

Case 1b In case 1b we use the same arguments as in cases 1c and 2a described above, with the only difference that $v_2(F) > a - 2c$. In particular,

$$\begin{aligned} v_2(A_2) &= v_2(A_2^*) + v_2(y) + v_2(F) + zc \\ &= v_2(A_2^*) + c + v_2(F) + zc > v_2(A_2^*) + a - c + zc \end{aligned}$$

$$\begin{aligned} v_2(A_1) &= v_2(A_1^*) + v_2(x) + zc \\ &= v_2(A_1^*) + a + zc \leq v_2(A_2^*) + a + zc \end{aligned}$$

Thus, we end up with an EFX allocation.

Case 1a In case 1a by considering the constraint defined in equation 2.1 we have that $a - b \geq v_2(F) \geq a - b - c$. After p_1 is unfrozen, we can have that there are still items valued as b from the perspective of p_2. In this case we have that the items obtained after the point where p_1 is unfrozen, have to be divided in $\lambda_{b,1}$, $\lambda_{c,1}$, $\lambda_{b,2}$ and $\lambda_{c,2}$ as the items that have value b and c for p_2, and are obtained by players p_1 and p_2 respectively. So we have the following considering that the non-frozen player has the precedence in the last iteration

$$\lambda_{b,1} = \begin{cases} \lambda_{b,2} & \implies \lambda_{c,1} = \begin{cases} \lambda_{c,2} \\ \lambda_{c,2} - 1 \end{cases} \\ \lambda_{b,2} - 1 & \implies \lambda_{c,1} = \begin{cases} \lambda_{c,2} \\ \lambda_{c,2} + 1 \end{cases} \end{cases}$$

The worst case to consider is when $\lambda_{b,1} = \lambda_{b,2} = \lambda_b$ and $\lambda_{c,1} = \lambda_{c,2} = \lambda_c$. In this case we have that A_1 is formed by A_1^*, item x obtained in the iteration that causes p_1 to be frozen,

z_b and z_c as the items obtained after that he has been unfrozen with value b and c for p_2. Instead A_2 is formed by A_2^*, item y obtained in the iteration that causes p_1 to be frozen, F the set of items obtained while p_1 was frozen and z_b and z_c as the items obtained after that he has been unfrozen respectively with value b and c for p_2. So we can write that

$$\begin{aligned} v_2(A_2) &= v_2(A_2^*) + v_2(y) + v_2(F) + z_b b + z_c c \\ &= v_2(A_2^*) + b + v_2(F) + z_b b + z_c c > v_2(A_2^*) + a - c + z_b b + z_c c \end{aligned}$$

$$\begin{aligned} v_2(A_1) &= v_2(A_1^*) + v_2(x) + z_b b + z_c c \\ &= v_2(A_1^*) + a + z_b b + z_c c \leq v_2(A_2^*) + a + z_b b + z_c c \end{aligned}$$

In the above equations I have considered that there are enough items with value c for p_2 to respect the constraint defined in 2.1, but if there are not enough of this items we have that $v_2(F) \geq a - 2b$ and the two following possible cases:

- $\min_{i \in A_1} v_2(i) = b$: in this case the value of A_2 for p_2 becomes:

  $$\begin{aligned} v_2(A_2) &= v_2(A_2^*) + v_2(y) + v_2(F) + z_b b \\ &= v_2(A_2^*) + b + v_2(F) + z_b b > v_2(A_2^*) + a - b + z_b b \end{aligned}$$

  that still is an EFX allocation since we have that from $v_2(A_1) \leq v_2(A_2^*) + a + z_b b$ we have to remove b.

- $\min_{i \in A_1} v_2(i) = c$: in this case we would have that this item i_c with value c for p_2 has been taken before that p_1 has been frozen, so in the same iteration p_2 took an item i_b with value at least b for p_2. So we can write the precedent equations as:

  $$\begin{aligned} v_2(A_2) &= v_2(A_2^* \setminus \{i_b\}) + v_2(i_b) + v_2(y) + v_2(F) + z_b b \\ &= v_2(A_2^* \setminus \{i_b\}) + b + b + v_2(F) + z_b b > v_2(A_2^*) + a + z_b b \end{aligned}$$

  $$\begin{aligned} v_2(A_1) &= v_2(A_1^* \setminus \{i_c\}) + v_2(i_c) + v_2(x) + z_b b \\ &= v_2(A_1^* \setminus \{i_c\}) + c + a + z_b b \leq v_2(A_2^*) + c + a + z_b b \end{aligned}$$

So also in case 1a the algorithm produces EFX allocations. For the next cases I will not consider the case in which we do not have enough items with value c for the non frozen player because as shown in the precedent part, we still have that the difference between the value for p_2 of A_1 and A_2 differs for at most the value of the item with lower value for p_2 in A_1 after that p_1 is unfrozen.

Now what remains to be shown is that the produced allocation by the algorithms is EFX, even if there exists a problematic assignment. In particular, we want to show that if after a problematic assignment the first run of the algorithm produces an allocation that is not EFX, then we can go back, change the assignment at thus point, run the algorithm from this point and onward, and obtain an EFX allocation. The possible problematic assignments are shown in Table 2.2:

p_1	a	b
p_2	b	c

p_1	a	b
p_2	a	b

Table 2.2. Problematic Assignments

p_1	**a**	b	b
p_2	b	c	**b**

Table 2.3. Assignment that shows that with the first problematic assignment we cannot have two frozen players or one player frozen twice

2.2.3 First Problematic Assignment

Let's start by showing that like in the precedent cases without problematic assignment, we have that only one player per run can be frozen. As first thing we can notice that in order to have a second freeze we must have that there is the assignment shown in Table 2.4, so we need to have an item valued by both players b after that a player has been frozen; but if this happens, than we had not followed the maximum matching in the first execution of the algorithm since we could have assigned the items as shown in Table **??** in bold. Indeed we have that $a + b > b + b$ and $a + b > a + c$.

Let's consider the first problematic assignment and that $a + c \geq b + b$: in this case by running the algorithm we obtain the following valuations for the two players

$$v_1(A_1) = v_1(A_1^*) + a + w_b b + w_c c$$
$$v_2(A_1) = v_2(A_2^*) + c + v_2(F_1) + w_b c + w_c c$$

where A_i^* is the set of items obtained in the iterations before the problematic assignment by p_i, w_b and w_c are respectively the number of items with value b and c obtained by p_1 after that he has been unfrozen and F_1 is the set of items obtained by p_2 while p_1 was frozen. If we consider that after that the frozen player is unfrozen, both player get the same number of items in order to have an EFX allocation we must respect the following two constraints since we assume that in the last iteration the non frozen player has the precedence.

$$v_1(A_1) \geq v_1(A_2) \tag{2.2}$$
$$v_2(A_2) \geq v_2(A_1) - c \tag{2.3}$$

In this case we have that the set of items A_1 will contain A_1^*, the item x such that $v_1(x) = a$ and $v_2(x) = b$ that has been obtained by p_1 in the iteration that causes p_1 to be frozen, w_b and w_c items obtained by p_1 after that has been unfrozen that have value b and c respectively for p_1. Instead A_2 will contain A_2^*, the item y such that $v_1(y) = b$ and $v_2(y) = c$ that has been obtained by p_2 in the iteration that causes p_1 to be frozen, w_b and w_c items obtained by p_1 after that has been unfrozen that have value b and c respectively for p_1. In order to respect the condition in 2.2 we must have that:

$$v_1(A_1) \geq v_1(A_2)$$
$$v_1(A_1^*) + v_1(x) + w_b b + w_c c \geq v_1(A_2^*) + v_1(y) + v_1(F_1) + w_b b + w_c c$$
$$a \geq b + v_1(F_1)$$

where we are considering $v_1(A_1^*) = v_1(A_2^*)$ cause this is the worst possible case since we always have that $v_1(A_1^*) \geq v_1(A_2^*)$ and this is due to the fact that these set of items are referred to the iterations before that a player has been frozen, so at each of these iterations we have that the item obtained by player 1 has a larger or equal value to the item obtained by player 2 for player 1 (the same holds for p_2). Instead in order to have that the condition in equation 2.3 we must have that

$$\begin{gathered} v_2(A_2) \geq v_2(A_1) - c \\ v_2(A_2^*) + c + v_2(F_1) + w_b c + w_c c \geq v_2(A_1^*) + b + w_b c + w_c c - c \\ c + v_2(F_1) \geq b - c \end{gathered}$$

where we can notice that the items relative to the counters w_b and w_c have value c for p_2 since he after the freeze can only take items with that value. This condition is always true since $c + v_2(F_1) = \left\lfloor \frac{b}{c} \right\rfloor c > b - c$. So the only possible problem is the first condition: $a \geq b + v_1(F_1)$. Since in F_1 there are the items taken by p_2 while p_1 is frozen, there can be items valued b by p_1; let's consider that we first assign to p_2 in F the items valued b by p_1 that have not yet assigned before the problematic assignment. In this case we can write

$$v_1(F_1) = kb + \left\lfloor \frac{b - c - kc}{c} \right\rfloor c = kb + (\left\lfloor \frac{b}{c} \right\rfloor - 1 - k)c$$

where k is the number of items valued b by p_1 that are in F_1 and $\lfloor \frac{b-c-kc}{c} \rfloor$ are the remaining items valued c by p_1 that p_2 needs to reach $v_2(F_1) = \lfloor \frac{b-c}{b} \rfloor c$. So we can write the condition in equation 2.2 as $a \geq b + kb + (\lfloor \frac{b}{c} \rfloor - 1 - k)c$, that is equivalent to

$$k \leq \frac{a + c - b - \lfloor \frac{b}{c} \rfloor c}{b - c} \tag{2.4}$$

If we change the problematic assignment we have that we will break the maximum matching rule and we will freeze p_2 rather than p_1. In this case we are considering w_b and w_c as the number of items with value b and c for p_1 respectively that p_1 take after that p_2 has been unfrozen, so we have that the valuation for the two players will be

$$\begin{aligned} v_1(A_1) &= v_1(A_1^*) + b + v_1(F_2) + w_b b + w_c c \\ v_2(A_1) &= v_2(A_2^*) + b + w_b c + w_c c \end{aligned}$$

The conditions in order to have an EFX allocation now become the following ones, since in this case p_1 will have the precedence in the last iteration:

$$v_1(A_1) \geq v_1(A_2) - c \tag{2.5}$$
$$v_2(A_2) \geq v_2(A_1) \tag{2.6}$$

In this case we have that A_1 will contain A_1^*, the item y obtained in the iteration in which we freeze p_2 such that $v_1(y) = b$ and $v_2(y) = c$, F_2 as the set of items obtained by p_1 while p_2 is frozen and w_b and w_c items obtained by p_1 after that p_2 has been unfrozen that have value b and c respectively for p_1. Instead A_2 will contain A_2^*, the item x obtained in the iteration in which we freeze p_2 such that $v_1(x) = a$ and $v_2(x) = b$ and w_b and w_c items obtained by p_2 after that has been unfrozen that have value b and c respectively for p_1. So the first condition is equivalent to

$$\begin{gathered} v_1(A_1) \geq v_1(A_2) - c \\ v_1(A_1^*) + v_1(y) + v_1(F_2) + w_b b + w_c c \geq v_2(A_1^*) + v_1(x) + w_b b + w_c b - c \\ b + v_1(F_2) \geq a - c \end{gathered}$$

That is always true since $v_1(F_2) \geq a - b - c$ as defined in equation 2.1. So now we have to check the condition for which p_2 is EFX towards p_1.

$$\begin{gathered} v_2(A_2) \geq v_2(A_1) \\ v_2(A_2^*) + v_2(x) + w_b c + w_c c \geq v_2(A_1^*) + v_2(y) + v_2(F_2) + w_b c + w_c c \\ b \geq c + v_2(F_2) \end{gathered}$$

We can notice that $v_2(F_2) = kc + \lfloor \frac{a-b-kb}{c} \rfloor c$ since, as said before, k is the minimum number of items that are surely present with value b for p_1 after the problematic assignment, so the other values that are used to reach the constraint $v_1(F_2) \geq a - b - c$ are in the worst case items with value c for p_1. We can also notice that all these items have value c for p_2 since he already took an item with value c, so there are no other items with value b or a. So we have that the condition in equation 2.6 becomes

$$\begin{aligned} v_2(A_2) &\geq v_2(A_1) \\ b &\geq c + v_2(F_2) \\ b &\geq c + kc + \left\lfloor \frac{a-b-kb}{c} \right\rfloor c \\ b &\geq c + kc + a - b - kb \\ \frac{a+c-2b}{b-c} &\leq k \end{aligned}$$

So the second condition in order to have an EFX allocation by changing the problematic assignment is

$$k \geq \frac{a+c-2b}{b-c} \tag{2.7}$$

As we can see, equation 2.4 and 2.7 are complementary on integer values of k. So if the algorithm produces a non EFX allocation, by changing the assignment we surely obtain an EFX allocation in the first problematic assignment of table 2.2.

2.2.4 Second Problematic Assignment

In order to show that also in this case we obtain an EFX allocation, I will first consider the case of identical valuation functions since is easier to deal with and than I will show that we can reduce a general case to this one. This reduction will come because I will show that in the case of non identical valuation function, by freezing one of the two players, we can obtain that the value for the frozen player of the set obtained by the non frozen one is lower or equal than the value obtainable in the case of identical valuation functions. We can easily note that with the second problematic assignment we could have two freezes: one caused by this problematic block and one caused than by the block shown in Table

p_1	b	c
p_2	b	c

Table 2.4. Second block that can cause a freeze when we have the second problematic block

2.4. In this part I am going to use $F_{i,j}$ in order to describe the set of items obtained by the player who took item with value j while the other player i is frozen and also

$$k = a \mod b$$
$$l = b \mod c$$
$$w = k \mod c$$

Identical Valuation Functions Let's consider the case in which we have identical valuation functions and we have that we freeze first p_1 and then p_2. In this case we can ignore the items that the players took in the iterations in which no player is frozen and the ones that do not lead to freeze one player because in each iteration each player has the same value for both the items assigned. So by considering only the iterations in which a player is frozen and the two that lead to freeze one player we can consider the valuations of each player set as

$$v(A_1) = a + c + v(F_{2,c})$$
$$v(A_2) = b + v(F_{1,b}) + b$$

where $v(\cdot) = v_1(\cdot) = v_2(\cdot)$. These values comes from the fact that in A_1 we have the item valued as a that leads to freeze p_1, c is the value of the item obtained in the iteration in which we freeze p_2 and $F_{2,c}$ is the set of items obtained when p_2 is frozen since p_1 took the item with value c. Instead in A_2 we have the item with value b obtained in the iteration in which p_1 is frozen, $F_{1,b}$ that is the set of items obtained by p_2 while p_1 is frozen since p_2 took the item with value b and b that is the value obtained by p_2 before being frozen. We can write that

$$v(F_{2,c}) + c = \left\lfloor \frac{b}{c} \right\rfloor c = b - (b\%c) = b - l \tag{2.8}$$

$$v(F_{1,b}) + b = \left\lfloor \frac{a}{b} \right\rfloor b = a - (a\%b) = a - k \qquad \textit{if } k = a\%b < c \tag{2.9}$$

$$v(F_{1,b}) + b = \left\lfloor \frac{a}{b} \right\rfloor b + \left\lfloor \frac{a - \lfloor \frac{a}{b} \rfloor b}{c} \right\rfloor c \tag{2.10}$$

$$= a - k + \left\lfloor \frac{a - a + k}{c} \right\rfloor c = a - k + \left\lfloor \frac{k}{c} \right\rfloor c \tag{2.11}$$

$$= a - k + k - k\%c = a - w \qquad \textit{if } k = a\%b \geq c \tag{2.12}$$

where in equation 2.9 we are considering that, since $a\%b < c$ than by taking $\lfloor \frac{a}{b} \rfloor$ items with value b we reach the bound $a - b - c$, while in equation 2.9 we need to take some items with value c in order to reach it. We can also notice that since we are considering that there is a second freeze, there is a number of items with value b greater or equal to $\lfloor \frac{a}{b} \rfloor$.

- if $k < c$ we have that

$$v(A_1) = a + c + v(F_{2,c}) \qquad v(A_2) = b + v(F_{1,b}) + b$$
$$v(A_1) = a + b - l \qquad v(A_2) = a - k + b$$

In the case considered above both players get the same number of items after that the last player has been unfrozen, in this case we can see that the allocation is EFX:

$$v(A_1) \geq v(A_2) - c \qquad -l \geq -k - c$$
$$v(A_2) \geq v(A_1) - c \qquad -k \geq -l - c$$

that is true since $k < c$ and $l < c$. Instead in the case in which we have an item that is taken by a player and not by the other, by assigning it correctly, we can obtain an EFX allocation: if we assign that item to p_1 we have that the condition to have an EFX allocation is

$$v(A_1) \geq v(A_2) - c \qquad -l \geq -k - c$$
$$v(A_2) \geq v(A_1) \qquad -k \geq -l$$

that is equivalent to have only the condition $k \leq l$. While if we assign it to p_2 the condition to have an EFX allocation is

$$v(A_1) \geq v(A_2) \qquad -l \geq -k$$
$$v(A_2) \geq v(A_1) - c \qquad -k \geq -l - c$$

that is equivalent to have the condition $l \leq k$.

- if $k \geq c$ we have that

$$v(A_1) = a + c + v(F_{2,c}) \qquad v(A_2) = b + v(F_{1,b}) + b$$
$$v(A_1) = a + b - l \qquad v(A_2) = a - w + b$$

In the case considered above both players get the same number of items after that the last player has been unfrozen, in this case we can see that the allocation is EFX:

$$v(A_1) \geq v(A_2) - c \qquad -l \geq -w - c$$
$$v(A_2) \geq v(A_1) - c \qquad -w \geq -l - c$$

that is true since $w < c$ and $l < c$ Instead in the case in which we have an item that is taken by a player and not by the other, by assigning it correctly, we can obtain an EFX allocation: if we assign that item to p_1 we have that the condition to have an EFX allocation is

$$v(A_1) \geq v(A_2) - c \qquad -l \geq -w - c$$
$$v(A_2) \geq v(A_1) \qquad -w \geq -l$$

that is equivalent to have only the condition $w \leq l$. While if we assign it to p_2 the condition to have an EFX allocation is

$$v(A_1) \geq v(A_2) \qquad -l \geq -w$$
$$v(A_2) \geq v(A_1) - c \qquad -w \geq -l - c$$

that is equivalent to have the condition $l \leq w$.

Non Identical Valuation Functions In the precedent paragraph I have shown that with identical valuation functions, also in the case of two freezes, we still can obtain an EFX allocation by correctly assigning the last item. In this section I am going to show that this holds also for non identical functions. I am going to show that this holds by showing that we have one of the two following conditions

- if we first freeze p_1 and than p_2 we have that $v_1(F_{1,b}) \leq v_2(F_{1,b})$
- if we first freeze p_2 and than p_1 we have that $v_2(F_{2,b}) \leq v_1(F_{2,b})$

Let's consider without loss of generality that the first condition holds, than after that p_1 is unfrozen, he will not envy p_2 since $a - b \geq v_2(F_{1,b}) \geq v_1(F_{1,b})$. Than in the iterations before the freeze of p_1 and the ones between the one in which we unfroze p_1 and the one in which we freeze p_2 we have that since no one is frozen, the value of the items obtained by p_1 is higher or equal to the value of the items taken by p_2 for p_1 (so in the worst case the two values will be the same). Than we have the second freeze, after which all the items are valued c by both players, so as we can see the worst case is the one in which we have identical valuation for p_1, this holds also for p_2 cause in the iterations before that p_1 is frozen and in the ones between the one in which we unfreeze p_1 and the one in which we freeze p_2, the items obtained by p_2 have an higher or equal value to the ones obtained by p_1 for p_2, and after the second freeze, all the items have value c for both players. So also for p_2 the worst case in which we have the two freezes is the one with the identical valuation functions. We can say the same things inverting p_2 and p_1 if the second condition holds. So we can reduce this case to the identical values functions case.

Let's consider the case in which we freeze first p_1, then p_2 will take $F_{1,b}$ items while p_1 is frozen. Of these items we can have three types of items:

1. item valued by both players as b
2. item valued by player p_1 as c and by p_2 as b
3. item valued by player p_1 as b and by p_2 as c

The same happens when we have that we freeze p_2 and than p_1. Let's consider that in both cases we use the same number of items of type 1: $t_{b,b}$, that of type 2 we have $t_{c,b}$ items and of type 3 we have $t_{b,c}$ items. So we have that in t_{**} the first letter in the subscript is the value for p_1, while the second one is the value for p_2. When freezing p_1 we can write that

$$\begin{aligned} v_1(F_{1,b}) &= t_{bb}b + t_{cb}c + \left\lfloor \frac{a-b-yb-vb}{c} \right\rfloor b \\ &= t_{bb}b + t_{cb}c + \hat{t}_{bc}b \\ v_2(F_{1,b}) &= t_{bb}b + t_{cb}b + \left\lfloor \frac{a-b-yb-vb}{c} \right\rfloor c \\ &= t_{bb}b + t_{cb}b + \hat{t}_{bc}c \leq a-b \end{aligned}$$

where $\hat{t}_{b,c} \leq t_{b,c}$ is the number of items of type 3 that p_2 needs to achieve the constraint

$v_2(F_{1,b}) \geq a - b - c$. In this case in order to have $v_1(F_{1,b}) \leq v_2(F_{1,b})$ we must have that

$$\begin{gathered} v_1(F_{1,b}) \leq v_2(F_{1,b}) \\ t_{bb}b + t_{cb}c + \hat{t}_{bc}b \leq t_{bb}b + t_{cb}b + \hat{t}_{bc}c \\ \hat{t}_{bc}(b-c) \leq t_{cb}(b-c) \end{gathered}$$

that is surely true when $s \leq v$ since $b > c$.
Instead if we freeze p_2 we have that

$$\begin{aligned} v_1(F_{2,b}) &= t_{bb}b + t_{bc}b + \left\lfloor \frac{a-b-yb-sb}{c} \right\rfloor c \\ &= t_{bb}b + t_{bc}b + \hat{t}_{cb}c \\ v_2(F_{2,b}) &= t_{bb}b + t_{bc}c + \left\lfloor \frac{a-b-yb-sb}{c} \right\rfloor b \\ &= t_{bb}b + t_{bc}c + \hat{t}_{cb}b \leq a - b \end{aligned}$$

where $\hat{t}_{c,b} \leq t_{c,b}$ is the number of items of type 2 that p_1 needs to achieve the constraint $v_1(F_{2,b}) \geq a - b - c$. In this case in order to have $v_2(F_{2,b}) \leq v_1(F_{2,b})$ we must have that

$$\begin{gathered} v_2(F_{2,b}) \leq v_1(F_{2,b}) \\ t_{bb}b + t_{bc}c + \hat{t}_{cb}b \leq t_{bb}b + t_{bc}b + \hat{t}_{c,b}c \\ \hat{t}_{cb}(b-c) \leq t_{bc}(b-c) \end{gathered}$$

that is surely true when $t_{cb} \leq t_{bc}$ since $b > c$.
So depending on t_{cb} and t_{bc} we can choose the player to freeze first and in the worst case we will obtain that the value of the set of items obtained by the unfrozen player for the frozen one is equal to the value for the unfrozen player. This worst case is leads to the same results obtained when we have identical valuation functions, so also if we have two freezes, than we still can obtain an EFX allocation by correctly assign the last item and by going back to the first freeze if we do not obtain an EFX allocation.

No double freeze In the case in which we have that there is the second problematic assignment and no second freeze, we can see that we still obtain an EFX allocation by considering that in the precedent paragraph we have shown that in a general case one of the following two cases is true

- if we first freeze p_1 and than p_2 we have that $v_1(F_{1,b}) \leq v_2(F_{1,b})$
- if we first freeze p_2 and than p_1 we have that $v_2(F_{2,b}) \leq v_1(F_{2,b})$

Let's consider that the first condition is true, than we have that when p_1 is unfrozen he will not envy p_2. If after that p_1 is unfrozen there are no freezes, than p_1 will take items with value higher or equal to the one obtained by p_2 for p_1, so at the end he will be EFX towards p_2 because at most p_2 will take the last item that will have the lower value for p_1 among the ones obtained by p_2. Instead for what concerns p_2, after that p_1 is unfrozen, he will have that $v_2(A_2) \geq v_2(A_1) - c$ because of the constraint described in equation 2.1. In the iterations after that p_1 is unfrozen, for p_2 holds the same thing that holds for p_1 with the only exception that p_2 has precedence in the last iteration, so also p_2 will be EFX towards p_1. So also in this case we obtain an EFX allocation.

2.2.5 Algorithm Cost

With respect to the original algorithm that worked on two values valuation functions, we have that the number of steps the algorithm does increases because of the fact that in case of non EFX allocation we have to go back to the iteration with one of the two problematic assignments and run again the algorithm. We could have also that the first assignment of the algorithm is a problematic assignment and this would mean that we could have to run the double of the steps required by the original algorithm.

2.2.6 Unify Non Problematic Assignment Proofs

In this section I am going to give a unified proof that in the case of non problematic assignments we can obtain EFX allocation by following only the new rules for allocate items while a player is frozen. Let's consider the fact that A_1^* and A_2^* are respectively the set of items obtained by p_1 and p_2 before the freeze and that we freeze player p_1 in an iteration in which he took an item x and p_2 took an item y, moreover let's consider that p_2 took a set of items F while p_1 is frozen. It's easy to see that since all the remaining items for the frozen player p_1 have value c for him, than he will be EFX towards p_2 in the end, because the items taken by p_2 will have higher or equal value to c and so $v_2(F) + v_2(y) \geq v_1(F) + v_1(y)$ and, since $v_1(x) \geq v_2(x) \geq v_2(F) + v_2(y)$ we have that $v_1(x) \geq v_1(y) + v_1(F)$. For what concerns p_2 we can write for all the non problematic cases the following equation because of the definition of the items assigned while a player is frozen:

$$v_2(x) - v_2(y) \geq v_2(F) \geq v_2(x) - v_2(y) - v_2(l_1)$$

where l_1 is the item with lower value for p_2 in the set A_1 at the end of the algorithm. By defining $\hat{A}_i$ as the set of items obtained by player i after that p_1 has been unfrozen, we can notice that $v_2(\hat{A}_2) \geq v_2(\hat{A}_1)$ since there are no other freezes and because the last iteration p_2 has precedence. So, because of we have that $A_2 = A_2^* \cup \{y\} \cup F \cup \hat{A}_2$ and that $A_1 = A_1^* \cup \{x\} \cup \hat{A}_1$ we can write

$$\begin{aligned} v_2(A_2) &= v_2(A_2^*) + v_2(y) + v_2(F) + v_2(\hat{A}_2) \\ &\geq v_2(A_2^*) + v_2(x) - v_2(l_1) + v_2(\hat{A}_2) \end{aligned}$$

$$\begin{aligned} v_2(A_1) &= v_2(A_1^*) + v_2(x) + v_2(\hat{A}_1) \\ &\leq v_2(A_2^*) + v_2(x) + v_2(\hat{A}_2) \end{aligned}$$

That clearly means that the allocation is EFX also for p_2 for the definition of l_1. In the precedent proof I have considered that in the case 1a there are enough items with value c for p_2 to reach the constraint described in 2.1, but if there are not enough of this items we have that $v_2(F) \geq a - 2b$ and the two following possible cases:

- $\min_{i \in A_1} v_2(i) = b$: in this case the value of A_2 for p_2 becomes:

$$\begin{aligned} v_2(A_2) &= v_2(A_2^*) + v_2(y) + v_2(F) + v_2(\hat{A}_2) \\ &= v_2(A_2^*) + b + v_2(F) + v_2(\hat{A}_2) > v_2(A_2^*) + a - b + v_2(\hat{A}_2) \end{aligned}$$

  that still is an EFX allocation since we have that from $v_2(A_1) \leq v_2(A_2^*) + a + v_2(\hat{A}_2)$ we have to remove an item valued b for the definition of EFX.

- $\min_{i \in A_1} v_2(i) = c$: in this case we would have that this item i_c with value c for p_2 has been taken before that p_1 has been frozen, because we are in the case in which the non frozen player has not enough items valued c to reach the constraint defined in 2.1, so such items is surely not assigned after that p_1 is unfrozen. By considering that in the same iteration in which p_1 took the item i_c p_2 took an item i_b with value at least b for p_2 (because this iteration is before the freeze of p_1 so there are still items valued at least b by p_2). So we can write the precedent equations as:

$$\begin{aligned} v_2(A_2) &= v_2(A_2^* \setminus \{i_b\}) + v_2(i_b) + v_2(y) + v_2(F) + v_2(\hat{A}_2) \\ &= v_2(A_2^* \setminus \{i_b\}) + b + b + v_2(F) + v_2(\hat{A}_2) > v_2(A_2^*) + a + v_2(\hat{A}_2) \end{aligned}$$

$$\begin{aligned} v_2(A_1) &= v_2(A_1^* \setminus \{i_c\}) + v_2(i_c) + v_2(x) + v_2(\hat{A}_1) \\ &= v_2(A_1^* \setminus \{i_c\}) + c + a + v_2(\hat{A}_1) \leq v_2(A_2^*) + c + a + v_2(\hat{A}_2) \end{aligned}$$

  Because $A_2 = A_2^* \setminus \{i_b\} \cup \{i_b\} \cup \{y\} \cup F \cup \hat{A}_2$ and $A_1 = A_1^* \setminus \{i_c\} \cup \{i_c\} \cup \{x\} \cup \hat{A}_1$

So also in case in which there are not enough items with value c for the frozen player to reach the constraint defined in 2.1, we can obtain EFX allocation.

Chapter 3

Three Players

3.1 Constraint on the Values

In the next pages I am going to show how to obtain an EFX allocation with three players and three values with the constraint that $c \geq a \mod b$. This constraint was not present in the discussion over two players, so why do I add it now? The answer is in how we deal with the fact that the frozen player envied the non frozen one in the two player case: it was enough to invert the problematic assignment. With the three-player case, this is no longer possible: let's consider the problematic assignment in which p_1 is frozen, p_2 and p_3 envies him and still can take other items valued b, in such a case both p_2 and p_3 by following the approach described in the case of two players could have to take items valued c before finishing the ones valued b if $c < a \mod b$, so we could have that not only p_1 starts to envy the non-frozen players, but also that the non-frozen player starts to envy each other and this is not solvable by changing the problematic assignment.

3.2 Redefinition of Problematic Assignment

For two players we have that only when the frozen player values remaining items as b, we could have a non EFX allocation by following the maximum matching assignments because in the end the frozen player could envy the non frozen player. In order to see what happens with three players we have to consider the different cases in which the frozen player values all the remaining items as c, let's consider that during the assignments that leads to freeze player p_1 he took item x, p_2 took item y and p_3 took item z:

- $v_1(x) = a, v_2(x) = a, v_2(y) = b, v_3(x) = a, v_3(z) = c$: in this case p_3 needs exactly $\lfloor \frac{a-c}{c} \rfloor$ items that have value c for p_1 and p_3, so p_1 when unfrozen will not envy p_3 (we have to notice that $v_1(z) = c$ otherwise we would have assigned x to p_3 and z to p_1 since $a + b + b > a + b + c$). Instead p_2 needs in the worst case (in the case in which he values all the remaining items as c) $\left\lfloor \frac{a-b}{c} \right\rfloor$ items, that have value lower or equal to $a - b - c$ for p_1, so also in this case p_1 will not envy p_2.

- $v_1(x) = a, v_2(x) = b, v_2(y) = c, v_3(x) = b, v_3(z) = c$: in this case both p_2 and p_3 have to take $\left\lfloor \frac{b-c}{c} \right\rfloor$ items. We have to see if $\left\lfloor \frac{b-c}{c} \right\rfloor \leq \lfloor \frac{a-w}{c} \rfloor$ holds where $w = \max\{v_1(y), v_1(z)\}$. If $w = c$, than surely the condition above holds, instead in the case of $w = b$ let's say that $v_1(y) = b$, than we have that $v_1(x) + v_2(y) \geq v_1(y) + v_2(x)$ because of the maximum matching rule, so $a + c \geq 2b$. So from the precedent constraint we can write that $a - b \geq b - c$, so we have that $\left\lfloor \frac{a-b}{c} \right\rfloor \geq \left\lfloor \frac{b-c}{c} \right\rfloor$. So in both cases, we have that the number of items obtained by the frozen player is lower to the ones needed to p_1 to envy one of the other players when unfrozen.

- $v_1(x) = b, v_2(x) = b, v_2(y) = c, v_3(x) = b, v_3(z) = c$: in this case the frozen player will take $\left\lfloor \frac{b-c}{c} \right\rfloor$ items, so p_1 will envy no player when unfrozen because $v_1(y) = v_1(z) = c$ for the maximum matching assignment.

3.3 We Cannot Use Maximum Matching

In the case of three players we cannot use the maximum matching technique to assign the items while a player is frozen in one of the problematic assignments because we cannot control the value for the frozen player of both the players. In table 3.1 there is an example that shows an allocation in bold obtained by choosing the items with the maximum matching technique while p_1 is frozen. As we can see also if we invert the assignment by freezing p_2 we can obtain the same assignment that leads the frozen player to envy p_3. In this particular case, by considering $a = 400, b = 100$ and $c = 40$, in the end we have that $v_1(A_1) = a + c = 440$ while $v_1(A_3) - \min_{x \in A_3} v_1(x) = 5b + c - c = 500$, so we obtain a non EFX allocation.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}
p_1	**a**	b	b	b	c	b	c	b	c	b	c	c	c	**c**
p_2	a	**b**	b	b	**b**	b	**b**	c	**c**	c	**c**	c	**c**	c
p_3	b	b	**b**	**b**	b	**b**	b	**b**	c	**b**	c	**c**	c	c

Table 3.1. An assignment that shows the problem of using maximum matching while a player is frozen because of a problematic assignment

3.4 First Problematic Assignment

				x	y	z	w
p_1	**a**	b	c	b	b	c	c
p_2	a	**b**	c	b	c	b	c
p_3	b	c	**c**	c	c	c	c

Table 3.2

In the table 3.2 the letters above each column are used in order to represent the type and also to represent the number of such items . Let's consider that we assign the items in bold, than we have to consider different cases:

- $w \geq \left\lfloor \frac{b}{c} \right\rfloor - 1$: in this case we can assign $\left\lfloor \frac{b}{c} \right\rfloor - 1$ items of type w to p_3 so that he does no longer envies p_1. Now we have the following situation for p_1:

$$v_1(A_2) = v_1(A_2^*) + b$$
$$v_1(A_3) = v_1(A_3^*) + \left\lfloor \frac{b}{c} \right\rfloor c \leq v_1(A_3^*) + b$$

so for p_1 we can threat the other two players in the same manner. Now we assign to p_2 and p_3 items of type x, z and y. Since $c \geq a\%b$ p_2 needs only items of value b to reach the constraint so we can consider the following cases:

- $\lfloor \frac{x+z}{2} \rfloor \geq \left\lfloor \frac{a-b}{b} \right\rfloor$: in this case is enough to let take the same number of items of each type to both p_2 and p_3 till reaching $\left\lfloor \frac{a-b}{b} \right\rfloor$ items. It is easy to see that p_1 will not envy p_2 or p_3 since $a - b \geq v_2(F_2) \geq v_1(F_2)$.
- in the case in which the condition above does not hold, than we have that p_2 and p_3 need to take $\hat{y}$ items of type y and this is not a problem if

$$\begin{cases} a - b \geq \lfloor \frac{x+z}{2} \rfloor b + \hat{y}c \\ a - b \geq \lfloor \frac{x}{2} + 1 \rfloor b + \lfloor \frac{z}{2} - 1 \rfloor c + \hat{y}b \end{cases}$$

  So we have a non EFX allocation if $\hat{y} > z$, in this case is enough to invert the assignment to the one shown in table 3.3 since is like considering p_1 inverted with p_2, so exchanging y with z.

p_1	a	**b**	c
p_2	**a**	b	c
p_3	b	c	**c**

Table 3.3

- $w < \left\lfloor \frac{b}{c} \right\rfloor - 1$: in this case we will not assign all the items of type w to p_3. So in this case we have that p_1 envies no one, p_2 envies only p_1 and the same p_3. In order to remove the fact that p_2 and p_3 still envy p_1 we need to assign to p_2 $\left\lfloor \frac{a-b}{b} \right\rfloor$ items of type x or z and other $\left\lfloor \frac{a-b-\lfloor \frac{x+z}{2} \rfloor b}{c} \right\rfloor$ items of type y or w if the items of type x and z are not enough; instead p_3 needs to obtain $\left\lfloor \frac{b}{c} \right\rfloor - 1$ items of any type. So let's consider the following cases:
  - $\lfloor \frac{x+z}{2} \rfloor \geq \left\lfloor \frac{a-b}{b} \right\rfloor$ and $\lfloor \frac{x+z}{2} \rfloor \geq \left\lfloor \frac{b}{c} \right\rfloor - 1$: in this case by assigning to p_2 and p_3 the items of type x and z till p_2 does not envy p_1 leads to an EFX allocation cause for p_1 the value obtained by the two players is lower or equal to $a - b$ as for p_2, for p_2 the obtained values is larger or equal to $a - b - c$ and for p_3 we have obtained the required number of items.
  - $\lfloor \frac{x+z}{2} \rfloor \geq \left\lfloor \frac{a-b}{b} \right\rfloor$ and $\lfloor \frac{x+z}{2} \rfloor < \left\lfloor \frac{b}{c} \right\rfloor - 1$: in this case we do not always obtain an EFX allocation because in order to have p_3 to not envy p_1 we could have to give to p_3 too many items that have value b for p_1, so p_1 would envy him when unfrozen. We can consider the following two case:

* $y \geq \left\lfloor \frac{z'}{2} \right\rfloor$: in this case we can change the assignment to the one show in table 3.4. Than we can assign the items as follows: assign half of the x items to p_1 and half to p_2 and than assign $\frac{z'}{2}$ items of type z to p_2 and $\frac{z'}{2}$ items of type y to p_1. If x is odd, than also z' is, in this case we have to assign the x odd item to p_1 and the z' odd item to p_2 in order to give each player $\left\lfloor \frac{a-b}{b} \right\rfloor$ items that he values b. So we have assigned $\left\lfloor \frac{x+z'}{2} \right\rfloor$ items to each player and now both no longer envy p_3, moreover p_3 will not envy them since $\left\lfloor \frac{x+z'}{2} \right\rfloor < \left\lfloor \frac{b}{c} \right\rfloor - 1$.
* $y < \left\lfloor \frac{z'}{2} \right\rfloor$: in this case we can avoid to change the assignment since, if $y < \frac{z'}{2}$ than $y < \left\lfloor \frac{a-b-\frac{x}{2}}{b} \right\rfloor$. In this case we have to do the following assignment: assign to p_2 $\frac{x}{2}$ items of type x, y items of type z and than $\frac{z'}{2} - y + \left\lfloor \frac{b-c-\frac{x}{2}c-\frac{z'}{2}c}{c} \right\rfloor$ items of type z, instead we have to assign to p_3: $\frac{x}{2}$ items of type x, y items of type y and than $\frac{z'}{2} - y + \left\lfloor \frac{b-c-\frac{x}{2}c-\frac{z'}{2}c}{c} \right\rfloor$ items of type z. As we can see p_1 will not envy p_2 since

$$\begin{aligned} v_1(F_2) &= \frac{x}{2}b + \frac{z'}{2}c + \left\lfloor \frac{b-c-\frac{x}{2}c-\frac{z'}{2}c}{c} \right\rfloor c \\ &\leq a - 2b + b - c = a - b - c \end{aligned}$$

since we are considering that $y < \left\lfloor \frac{z'}{2} \right\rfloor$, so $\left\lfloor \frac{z'}{2} \right\rfloor \geq 1$ we have $\frac{x}{2}b \leq a - 2b$ and we also have that $\frac{z'}{2}c + \left\lfloor \frac{b-c-\frac{x}{2}c-\frac{z'}{2}c}{c} \right\rfloor c \leq b - c$. p_1 will also not envy p_3 as we can see by the next equation:

$$\begin{aligned} v_1(F_3) &= \frac{x}{2}b + yb + (\frac{z'}{2} - y)c \left\lfloor \frac{b-c-\frac{x}{2}c-\frac{z'}{2}c}{c} \right\rfloor c \\ &\leq a - 2b + b - c = a - b - c \end{aligned}$$

since, as before, $y < \frac{z'}{2}$, we have $\frac{x}{2}b + yb \leq a - 2b$ and we also have that $(\frac{z'}{2} - y)c \left\lfloor \frac{b-c-\frac{x}{2}c-\frac{z'}{2}c}{c} \right\rfloor c \leq b - c$. We can notice that if x is odd, than we still have the same conditions if we consider that we give the x odd item to p_3 and that $y < \left\lfloor \frac{a-b}{b} \right\rfloor - \lfloor \frac{x}{2} \rfloor - 1$, if this does not hold, than we can see that we obtain an EFX allocation by following the approach described in the precedent case.

p_1	a	**b**	c
p_2	a	b	**b**
p_3	**b**	c	c

Table 3.4

- $\lfloor \frac{x+z}{2} \rfloor < \left\lfloor \frac{a-b}{b} \right\rfloor$: in this case we must take also items of type y. Let's consider that we give y' items to the non frozen players of type y, than we must have that two or none between x, y' and z are odd because $x + y' + z$ has to be even in order to split the items between the two non frozen players. Let's deal with the possible case one per time:

* $\left\lfloor \frac{y'}{2} \right\rfloor \leq \lfloor \frac{z}{2} \rfloor$: in order to solve this problem we have to do the following assignments: to p_2 and p_3 $\lfloor \frac{x}{2} \rfloor$ items of type x, than we assign one item of the two types that have an odd number of items by giving the one (there is always at least one) with value b for p_2 to p_2, than we assign to each player $\lfloor \frac{z}{2} \rfloor$ items of type z and in the end we assign $\left\lfloor \frac{y'}{2} \right\rfloor$ items of type y to each player. In all cases, since the items of type z given to each player are more than the ones of type y, the value for p_1 of the value obtained by the two players will be lower or equal to the value obtained by p_2 that is lower or equal to $a - b$.
* $\left\lfloor \frac{y'}{2} \right\rfloor > \lfloor \frac{z}{2} \rfloor$: we can see that in this case we can do the same thing done before by exchanging p_1, p_2 and z and y'.

In all the above cases I have assumed that the number of items obtained by p_3 are greater or equal to $\left\lfloor \frac{b}{c} \right\rfloor - 1$ because when we assign to p_2 items with value c for him (items of type y and w), we are assigning them in order give him at least a value of $b - c$ (because we have the constraint $c \geq a \mod b$), so also considering only these items, p_3 took enough items to no longer envy p_1.

We can notice that for how we give the items to the two non frozen players, we will never have that one player envies the other, so we will never have to freeze a player in this phase. Instead after that the frozen player p_i is unfrozen, we could have another assignment that can lead to a freeze: in this case we can see that p_3 is not a cause of this freeze since all the remaining items have value c, so we will freeze one player among p_1 and p_2. If this happens, than we have to freeze the player that has not yet been frozen. Let's consider that in the first freeze we froze p_1, than when p_1 is unfrozen he is envy-free to the other two players, while for p_2 we have that $v_2(A_2) \geq v_2(A_1) - c$, so by freezing p_2 we ensure the fact that p_1 takes value lower than p_2 with a difference of at most a c, while p_2 takes items with larger or equal value to the ones obtained by p_1, so we are ensuring the fact that p_1 and p_2 are EFX.

3.5 Second Problematic Assignment

	i_1	i_2	i_3	z	w
p_1	**a**	b	b	b	c
p_2	b	**c**	c	c	c
p_3	b	c	**c**	c	c

Table 3.5

In this case is easy to see that p_2 and p_3 both need $\left\lfloor \frac{b-c}{c} \right\rfloor$ items of any remaining type. We can differentiate two cases by considering the number $\hat{z}$ of items of type z that we have to assign to p_2 and p_3 to reach the required number of items

- $\lfloor \frac{w}{2} \rfloor c + \hat{z}b \leq a - b$: where $\hat{z} = \left\lfloor \frac{b-c-\lfloor \frac{w}{2} \rfloor c}{c} \right\rfloor$, in this case is enough to assign to p_2 and p_3 first the items of type w, than the items of type z in equal values.

- $\lfloor \frac{w}{2} \rfloor c + \hat{z}b > a - b$: where $\hat{z} = \left\lfloor \frac{b-c-\lfloor \frac{w}{2} \rfloor c}{c} \right\rfloor$, in this case we have to invert the assignment so that we freeze player p_2 rather than p_1. Now we have that p_1 and p_3 will envy p_2 for different values. By assigning the items of type z to p_1 and of type w to p_3 till p_3 reaches $\left\lfloor \frac{b-c}{c} \right\rfloor$ items, we have an EFX allocation since
  - p_1 will not envy p_2 since he surely took more than $a - b$ value and equal or higher value than p_3 while p_2 was frozen,
  - p_2 will not envy the other players since they took at most $b - c$ value while p_2 was frozen,
  - p_3 will not envy p_1 since they take the same items and will not envy p_2 by considering that p_3 ha precedence over the last iteration assignment.

3.6 Third Problematic Assignment

				x	y	z	j	k	l	w	v
p_1	**a**	b	b	b	c	b	b	c	b	c	c
p_2	a	**b**	b	b	b	c	b	b	c	c	c
p_3	a	b	**b**	c	c	c	b	b	b	c	b

Let's assume that

$$x + y + k + j \geq j + k + l + v \geq x + z + j + l$$

so that the second player has the higher number of items that he values b, followed by p_3 and p_1. Now we can show that p_1 will never envy the other two players when he is unfrozen if we assign the items as follows till both player have reached $v_i(F_i) \geq a - b - c$: we assign first the items of type y and x to p_2, and the items of type v and l to p_3. Since we have assumed that $x + y + k + j \geq j + k + l + v$, we have $x + y \geq l + v$ so p_3 will finish first the items of type l and v, so while p_2 is still taking items of type x and y he will star taking items of type k and l. Now we can have two different cases:

- $x + y \geq l + v + j + k$: in this case p_3 will finish all the items that he values b (l, v, j and k) while p_2 is still taking items of type x and y. So after that p_3 has finished these items we will start assigning the same type of items to the two non frozen players in the following order x,y, w and z. In this case is easy to see that p_1 will not envy the non frozen players when he will be unfrozen because p_3 takes all the items that he values b and by the initial assumption we have that $j + k + l + v \geq x + z + j + l$, so p_3 will surely achieve $a - b - c$ before p_1 does it considering the bundle obtained by one of the two non frozen players.

- $x + y < l + v + j + k$: in this case p_2 will finish all the items of type x and y when there are still left other items of type j and k. We define to be k' and j' the items of type k and j respectively taken by p_3 while p_2 is still taking items of type x and y, and we define rem_k and rem_j to be $k - k'$ and $j - j'$ respectively. We will divide rem_k and rem_j equally among the two non frozen players and if the remaining j and k items are odd, we give that item to p_2 and we freeze him till p_3 obtains $\left\lfloor \frac{b-c}{c} \right\rfloor$ items among the remaining ones that are of type w and z. Both in case of a second freeze and in case of no second freeze while p_1 is frozen, we have that at most p_1 will envy one of the two other players and not both. In table 3.6 and 3.7 we can see the assignments done for the case in which we have a freeze and the case in which we have no freeze, where we consider rem_z to be the number of remaining items of type z after that p_2 has been unfrozen in the case of second freeze. We can see that for p_1 to envy both other players we must have that both $v_1(F_2) > v_2(F_2)$ and $v_1(F_3) > v_3(F_3)$ are true. In the following equations we can see that this is impossible in the case of second freeze while p_1 is frozen, but this can also consider the case in which we have no second freeze because is enough to have $rem_z = z$.

$$v_1(F_2) = xb + yc + \left\lfloor \frac{rem_j}{2} \right\rfloor b + \left\lfloor \frac{rem_k}{2} \right\rfloor c + b + \left\lfloor \frac{rem_z}{2} \right\rfloor b + \left\lfloor \frac{rem_w}{2} \right\rfloor c$$
$$v_2(F_2) = xb + yb + \left\lfloor \frac{rem_j}{2} \right\rfloor b + \left\lfloor \frac{rem_k}{2} \right\rfloor b + b + \left\lfloor \frac{rem_z}{2} \right\rfloor c + \left\lfloor \frac{rem_w}{2} \right\rfloor c$$

$$v_1(F_3) = vc + lb + j'b + k'c + \left\lfloor \frac{rem_j}{2} \right\rfloor b + \left\lfloor \frac{rem_k}{2} \right\rfloor c + c + (z - rem_z)b \left\lfloor \frac{rem_z}{2} \right\rfloor b + \left\lfloor \frac{rem_w}{2} \right\rfloor c$$
$$v_3(F_3) = vb + lb + j'b + k'b + \left\lfloor \frac{rem_j}{2} \right\rfloor b + \left\lfloor \frac{rem_k}{2} \right\rfloor b + c + (z - rem_z)c \left\lfloor \frac{rem_z}{2} \right\rfloor c + \left\lfloor \frac{rem_w}{2} \right\rfloor c$$

$$v_1(F_2) > v_2(F_2) \implies \left\lfloor \frac{rem_z}{2} \right\rfloor > y + \left\lfloor \frac{rem_k}{2} \right\rfloor$$
$$v_1(F_3) > v_3(F_3) \implies \left\lfloor \frac{rem_z}{2} \right\rfloor + z - rem_z > v + k - rem_k + \left\lfloor \frac{rem_k}{2} \right\rfloor$$

By summing the two inequalities we obtain

$$2\left\lfloor \frac{rem_z}{2} \right\rfloor + z - rem_z > y + v + k - rem_k + 2\left\lfloor \frac{rem_k}{2} \right\rfloor$$

that is impossible because of the initial assumption, indeed $z \leq k + v + y$. So p_1 will at most envy one of the other two players. This will happen only when the two non frozen player are not able to reach $a - b - c$ with the only items of type b because otherwise is immediate that the value for p_1 of the bundles obtained by the other two players while he was frozen is lower or equal to the value obtained by the two non frozen players that is lower or equal to $a - b - c$. So in the case in which p_1 envies one of the two other players we can swap the bundles obtained from the problematic iteration till the iteration in which we unfreeze p_1 of these two players.

Now let's consider first the case in which we had no swap after the unfreeze of p_1 and than the case in which we have it:

p_2	x	y			$\lfloor \frac{rem_k}{2} \rfloor$	$\lfloor \frac{rem_j}{2} \rfloor$	odd k or j	frozen	$\lfloor \frac{rem_w}{2} \rfloor$	$\lfloor \frac{rem_z}{2} \rfloor$	z odd
p_3	v	l	k'	j'	$\lfloor \frac{rem_k}{2} \rfloor$	$\lfloor \frac{rem_j}{2} \rfloor$	w or z	w, z	$\lfloor \frac{rem_w}{2} \rfloor$	$\lfloor \frac{rem_z}{2} \rfloor$	w odd

Table 3.6. Assignment in the case of second freeze while p_1 is frozen

p_2	x	y			$\lfloor \frac{rem_k}{2} \rfloor$	$\lfloor \frac{rem_j}{2} \rfloor$	$\lfloor \frac{w}{2} \rfloor$	$\lfloor \frac{z}{2} \rfloor$	z odd
p_3	v	l	k'	j'	$\lfloor \frac{rem_k}{2} \rfloor$	$\lfloor \frac{rem_j}{2} \rfloor$	$\lfloor \frac{w}{2} \rfloor$	$\lfloor \frac{z}{2} \rfloor$	w odd

Table 3.7. Assignment in the case of no second freeze while p_1 is frozen

- in the cases in which we had no swap after the unfreeze of p_1 we have the following situation

$$v_1(\hat{A}_1) + a \geq v_1(\hat{A}_2) + b + v_1(F_2)$$
$$v_1(\hat{A}_1) + a \geq v_1(\hat{A}_3) + b + v_1(F_3)$$

$$v_2(\hat{A}_2) + b + v_2(F_2) \geq v_2(\hat{A}_1) + a - c$$
$$v_2(\hat{A}_2) + b + v_2(F_2) \geq v_2(\hat{A}_3) + b + v_2(F_3)$$

$$v_3(\hat{A}_3) + b + v_3(F_3) \geq v_3(\hat{A}_1) + a - c$$
$$v_3(\hat{A}_3) + b + v_3(F_3) \geq v_3(\hat{A}_2) + b + v_3(F_2) - c$$

Where in the last inequality we have the last $-c$ only if we had a second freeze.

After this we have that in the last iteration we will assign the items with priority to p_2 and p_3, and among these two players with priority to the player who values more the item if we had no second freeze, or to p_3 if we had a second freeze for which we froze p_2. The precedent assertion works only when the last item has value c for p_1, since he is EF towards the other two players till the last iteration, or p_2 and p_3 did not took any item of type z while frozen (so in the case in which the last item is of type j, k, x, y, l and v) since in this case each item obtained by the frozen players while p_1 was frozen, has value higher or equal, for the player who took it, to the value that has for p_1, so in the case in which the last item values b for p_1 we can have two cases: $\min_{x \in A_i} v_1(x) = b$ or $\min_{x \in A_i} v_1(x) = c$, the first case is easy for the definition of EFX, the second case instead has more to argue. In the case in which $\min_{x \in A_i} v_1(x) = c$, we have that or this item has been obtained while p_1 was frozen, and this would imply that $v_1(F_i) \leq a - b - b + c = a - 2b + c$ so we can assign the last item to player i keeping p_1 EFX towards p_i, or is associated to an item that p_1 took with value b, so we can still assign the last item to p_i by keeping p_1 EFX towards p_i. Instead if the last item is of type z, than we could have assigned some items of type z to p_i, in this case we can't assert the same things that I have told before for the case in which $\min_{x \in A_i} v_1(x) = c$. Let's divide the cases based on the values of each player for it's set of items obtained from the iteration in which we freeze p_1 and let's call this set for a generic player p_i as $\bar{A}_i$:

- $v_2(\bar{A}_2) \geq v_2(\bar{A}_1)$ and $v_3(\bar{A}_3) \geq v_3(\bar{A}_1)$: in this case we have to assign the last item to p_1 and if the last items are two, if there was a second freeze, to the non frozen player, otherwise to one between p_2 and p_3. We can notice that since the item is valued c by p_2 and p_3 we obtain an EFX allocation.
- $v_2(\bar{A}_2) \geq v_2(\bar{A}_1)$ and $v_3(\bar{A}_3) \geq v_3(\bar{A}_1) - c$: in this case we can notice that since p_2 took more value than the value obtained by p_1 or he had to wait p_3 ($x + y \geq j + k + v + l$) or after the freeze he took some item with value b while in the same iteration p_1 took an item valued c by p_2; in both cases we have that there has not been a second freeze between p_2 and p_3, so we have that p_2 and p_3 are EF till now. If there are two last items, we assign them to p_3 and p_1 since p_2 is EF towards both of them, instead if is only one item and we have that $v_1(\bar{A}_3) + b - c \leq v_1(\bar{A}_1)$, we can assign the last item to p_3. The last case to consider is when we have only one last item and $v_1(\bar{A}_3) + b - c > v_1(\bar{A}_1)$, in this case we can swap the bundles of p_1 and p_3 and assign the last item to p_1. This because we will obtain $N_1 = \bar{A}_3, N_2 = \bar{A}_2$ and $N_3 = \bar{A}_1$ such that $v_1(N_1) + b - c \geq v_1(N_3)$ and $v_3(N_3) \geq v_3(N_1)$.
- $v_2(\bar{A}_2) \geq v_2(\bar{A}_1) - c$ and $v_3(\bar{A}_3) \geq v_3(\bar{A}_1)$: in this case hold the same things as before.
- $v_2(\bar{A}_2) \geq v_2(\bar{A}_1) - c$ and $v_3(\bar{A}_3) \geq v_3(\bar{A}_1) - c$: this case can be handled like the precedent one, but in this case we have that p_2 and p_3 could not be EF. If there is only one last item and we have that for at least one between p_2 and p_3 we have that $v_1(\bar{A}_i) + b - c \leq v_1(\bar{A}_1)$ than we assign the last item to p_i. If instead this does not hold, than we can exchange the bundles of p_1 and p_i where $i = \arg\max_{i \in \{2,3\}} v_1(\bar{A}_i)$. Than we can assign the last items to p_1 and p_j (not p_i). Assigning the last item also to p_j could provoke to not having p_i EFX towards p_j because of a second freeze in which we froze p_j, so in this case we change that assignment by freezing p_i and not p_j. We can notice that the value $v_1(\bar{A}_i)$ will only increase with this last swap, because p_i will take more items than before, so the exchange of bundles between p_1 and p_i done before will not change.

- Instead in the case in which we had to swap after the unfreeze of p_1 we have that

$$v_1(\hat{A}_1) + b + v_1(F_2) \geq v_1(\hat{A}_2) + a$$
$$v_1(\hat{A}_1) + b + v_1(F_2) \geq v_1(\hat{A}_3) + b + v_1(F_3)$$

$$v_2(\hat{A}_2) + a \geq v_2(\hat{A}_1) + b + v_2(F_2)$$
$$v_2(\hat{A}_2) + a \geq v_2(\hat{A}_3) + b + v_2(F_3)$$

$$v_3(\hat{A}_3) + b + v_3(F_3) \geq v_3(\hat{A}_1) + b + v_3(F_2) - c$$
$$v_3(\hat{A}_3) + b + v_3(F_3) \geq v_3(\hat{A}_2) + a - c$$

where the $-c$ in the last two rows is the first one because of in the case of second freeze while p_1 was frozen, we have freeze p_2 but now that bundle is of p_1 and the second one because p_2 now has the item valued a and p_3 has that $v_3(F_3) \geq a - b - c$. In this case we have that the precedence to take the last items is of p_3 because is the only non EF player towards the others.

In this case we can notice that p_1 and p_2 are EF towards the other two players and also that this case only happens when the two non frozen player start to take items of type z, so we can assert that the only two types of items that will be as last item are w and z. In the case of last item of type w is enough to assign it with precedence to p_3, instead in case of last item of type z, if there are two of such items, we can assign one to p_1 and one to p_3. Instead in the case of only one last item of type z we have to deal with the fact that p_3 needs to take the last item and also that that item values b for p_1. In the case in which in F_2 there is an item valued c by p_1, we exchange that item with the last item and than we give it to p_3. Now I'll show that the bundle obtained while p_1 was frozen by the player who exchanged it with him has at least one item that p_1 values c: let's start considering F_2, than if by absurd we consider that $rem_k = 0$ than we must have that $x + y \geq v + l + k$, but than also $y = 0$, so we would have that $x \geq v + l + k$ and this violates the first assumption that for which $k + y \geq z + l$ and so $k \geq z + l$ and also $k + v \geq x + z$, but before we wrote that $x \geq v + l + k$. Instead in the case of F_3 we have that to have no item valued c, we must have that $k = 0, v = 0$ and this violates the initial assumption for which $k + v \geq x + z$.

3.7 Problematic Assignments Where There Is a Player That Does Not Envy the Other

	d	e	f	g	j	k	y	x	v	l	z	w
p_1	b	b	c	c	b	c	c	b	c	b	b	c
p_2	b	c	b	c	b	b	b	b	c	c	c	c
p_3	a	a	a	a	b	b	c	c	b	b	c	c

Table 3.8. Types of items after a problematic assignment

In this section I am going to describe the approach used to obtain EFX allocation when we have a problematic assignment in which a player (p_3 without loss of generality) does not envy the item obtained by the other two players. In the beginning of the proof I will show the case in which each type of item has an even number of elements, than I will show that in the case in which this not hold, we can obtain the same characteristics with other assignments.

Even number of items Let's consider the case in which we have even number of items for each remaining type after the problematic assignment, in this case we can assign the items to p_3 by letting him choose the one with higher value for him and than assign the same type of item to the other non frozen player (p_2 without loss of generality). By doing so we have that $v_1(F_2) = v_1(F_3)$, so if when unfrozen p_1 envies p_3 than he envies also p_2. We have two cases now:

- when unfrozen $v_1(F_2) \leq a - b$: in this case we have that all players are EF towards each other except for p_2 that has $v_2(\bar{A}_1) \geq v_2(\bar{A}_2) > v_2(\bar{A}_1) - c$, where $\bar{A}_i$ is the set of items obtained by player i from the problematic assignment on.

- when unfrozen $v_1(F_2) > a - b$: in this case we have to invert the assignment and give $\bar{A}_1$ to p_2 and $\bar{A}_2$ to p_1 and after this we have that all the players are EF towards each other.

Now we have to deal with the fact that the player who has F_2 (p_2 in the first case and p_1 in the second case) did not took the items by first taking all the items with value b and than the ones with value c because he took always the same item took by p_3, so we could have some problems when assigning the last items. As first thing we can notice that we can avoid to give the last item to p_3 because he is EF towards p_2, p_1 and took the items in the right order, so we have to deal only on the case in which we have a single last item that we have to give to p_1 or p_2. The main problem raises when the last item is valued by both p_1 and p_2 as b. Now in order to show how to deal with this case I am going to consider the case in which there was no swap after that p_1 has been unfrozen, in the other case is enough to exchange p_1 with p_2. So if we have that $\min_{x \in A_2} v_1(x) = b$ than we can assign the item to p_2 and obtain an EFX allocation directly, instead in the case in which $\min_{x \in A_2} v_1(x) = c$ than we have to consider when the item g valued c by p_1 has been obtained: if it has been obtained before or after the problematic assignment, than surely p_1 took in the same iteration an item valued b or higher, so by assigning the last item to p_2 we still have an EFX allocation. In the case in which the item valued c by p_1 has been obtained when p_1 was frozen, than we have that the last item can be of two types:

- $g = (c, b, \star)$: in this case or we have that $\exists f = (b, c, \star) \in F_2$ or $v_1(F_2) \leq a - 2b + c$. In the latter case we can assign the last item directly to p_2 by keeping an EFX allocation, in the first case instead we can assign to p_1 f, while to p_2 we assign g. In this manner we decrease the value for p_1 and p_2 of the other player set of items of $b - c$, so we can assign to p_2 the last item. We can also notice that the value of A_2 will not increase for p_3 since the last item has surely lower or equal value to the item removed. Instead for what concerns p_1 for p_3, since p_3 did not envied him when p_1 was frozen, p_3 has $v_3(A_2) \leq v_3(A_3 \setminus F_3)$ and $v_3(F_3) \geq v_3(f)$, so also for p_3 this is still an EFX allocation.

- $g = (c, c, \star)$: this last case is the case in which all the items in A_2 valued c by p_1 value also for him c and have been obtained while p_1 was frozen. In this case we give all (or at most $\left\lfloor \frac{b}{c} \right\rfloor$) such type of items to p_1 from F_2 and assign the last item to p_2. If there are not enough such items we are still assigning them to p_1 and than we obtain that $\min_{x \in A_2} v_1(x) = b$, so we can assign the last item to p_2. If instead there are such items, we obtain an EFX allocation because the value for p_1 of A_2 increases at most of c. Also in this case for p_3 this is not a problem because the items assigned to p_1 from F_2 have still value lower or equal to $v_3(F_3)$.

non even number of items in the precedent paragraph I have shown that by giving the same items to p_3 and p_2 chosen by p_3 we obtain an EFX allocation in the end, the main characteristic of such assignments that lead to an EFX allocation is that $v_1(F_2) \geq v_1(F_3)$ and that p_2 and p_3 are EF. In this paragraph I am going to show ho to keep such constraints when we have no even number of items and, in the case in which is not possible, how to still obtain an EFX allocation. In the next part I am going to show how to deal with the most difficult cases, for all the ones that have not been discussed here, it means that or we can assign the odd g item to p_3 and we have another one f such that $v_2(f) \geq v_2(g)$ and $v_1(f) \leq v_1(g)$ or the opposite.

- d is odd:
  - $e, f, g, j, x = 0$: in this case we can assign to p_2 one item between k or z and one between k or y, while to p_3 we assign the odd d item. In this case I have to highlight that p_3 will not envy p_2 because is impossible to have $2b \geq a$, indeed in this case we would have that no item is taken while p_1 is frozen because $b \geq a - b$.
  - $e, f, g, j, x, l, z = 0$: in this case we have to make a manual assignments and not follow the algorithm described in this subsection. In this case we have that the only types of items that are left are d, k, y, w and v. In this case we can split the d items between p_2 and p_3 and than assign the odd d item to p_3, while one between k or y to p_2 (or in the case in which $k = y = 0$ we assign $\left\lfloor \frac{b}{c} \right\rfloor$ items between w and v and in the case in which these item have value higher of a for p_3, than invert such last assignment)
  - $e, f, g, j, x, k, y = 0$: in this case, since $f + k = 0$ we have that also $l + e = 0$ and so $l = e = 0$. So in this case we only have the following items: d, z, w and v moreover, since this case has to differ from the precedent one, we have that $z \neq 0$ (before we have also considered the case in which $l = z = k = y = 0$) so in this case we can freeze p_2 and not p_1 and assign to p_1 and p_3 each half of the items of type d; than we assign to p_3 the odd d item and to p_1 an item of type z, than we continue to assign items of type z to p_2 and of type v to p_3. When we unfreeze p_2, he will not envy the other two players.
- e is odd:
  - $z, x, l, g, j = 0$ in this case we can assign to p_2 the odd e item and an item of type y, while to p_3 we assign an item of type f and one of type v or w
  - also $y = 0$: in this case we assign to p_2 the odd e item and an item of type k, while to p_3 we have to assign an item of type f and one of type v (not w).
  - also $v = 0$: in this case we cannot follow the algorithm, but in this case we have very few types of items, so we can assign the items manually as follows: we start assigning to p_2 the items of type k (if there are, k in this case could be also 0) and f, while to p_3 we assign the items of type e and since we are considering that $f + k \geq l + e$ and that $l = 0$, than $k + f \geq e$ and so the $v_1(F_2) \leq v_1(F_3) \leq v_2(F_2) \leq a - b$. So when unfrozen p_1 will not envy p_2 or p_3. From this we can easily obtain an EFX allocation as described in the beginning of the subsection.
  - $z, x, l, g, j, f = 0$: in this case we have to assign the items manually in the following manner: first we assign the items of type d to both p_2 and p_3, than, since $f + k \geq l + e$ and so $k \geq e$, we assign the items of type k to p_2 and the items of type e to p_3, than we will assign the remaining items of type y

and v respectively to p_2 and p_3. As before, we have that $v_1(F_2) \leq v_1(F_3) \leq v_2(F_2) \leq a - b$ and so we can obtain an EFX allocation in the end.

- $l, v, k = 0$ and j is odd: in this case we can assign the odd j item to p_3 and $\left\lfloor \frac{b}{c} \right\rfloor$ items between the ones of type x, z, w, y to p_2. In this case we can notice that p_1 will envy more p_3 but we still have that $v_1(F_2) > v_1(F_3) - c$, so by swapping with p_2 and not p_3 we still obtain EFX allocation since in the end p_3 does not take items in the last iteration.
- $x, z, v = 0$ and l is odd: in this case we assign the odd l item to p_3 and $\left\lfloor \frac{b}{c} \right\rfloor$ items between the ones of type w, y to p_2. In this case holds the same thing said in the precedent point.
- $y, x = 0$ and k odd: in this case we assign the odd k item to p_3 and $\left\lfloor \frac{b}{c} \right\rfloor$ items between the ones of type z, w to p_2. In this case holds the same thing said in the precedent point.

Till now I have not considered the fact that, when p_2 lost an item valued a, since p_2 is not taking the items following their value for him, we have that cause of the presence of some items valued c b p_2 in F_2, we could reach a point in which by adding an item g we have that $v_2(F_2) \leq a - b - c$ and $v_2(F_2) + v_2(g) \geq a - b$ and $v_1(F_2) \leq a - b - c$ and $v_1(F_2) + v_1(g) \geq a - b$. If such thing happen, than we cannot invert the assignment and obtain an EF allocation because the bundle given to p_1 would have value higher than a for p_2. Such thing can happen after that p_2 took value b for himself and p_1. In order to solve such problem we can divide in the following cases:

- $\exists h = (b, c, \star) \in F_2$: in this case is enough to add g to F_2 and remove h from F_2, after this the value for p_2 of F_2 is increased by $b - c$ while remained constant for p_1, so we would have that $v_2(F_2) \geq a - b - c$ and $v_1(F_2) \leq a - b$, so we can unfreeze p_1 and move forward.
- $(c, b, \star) \in F_2$: in this case, by excluding the precedent case, we have that is impossible have that $v_2(F_2) \leq a - b - c$ and $v_2(F_2) + v_2(g) \geq a - b$ and $v_1(F_2) \leq a - b - c$ and $v_1(F_2) + v_1(g) \geq a - b$, because $v_2(F_2) = v_2(F_2 \setminus \{g\}) + b$ and $v_1(F_2) = v_1(F_2 \setminus \{g\}) + c$ and $v_2(F_2 \setminus \{g\}) \geq v_1(F_2 \setminus \{g\})$, so we have in the end that $v_2(F_2) - v_1(F_2) \geq b - c$
- only $(b, b, \star), (c, c, \star) \in F_2$: in this case we can remove the items that causes the shift between multiples of b and the value of F_2 for p_2 and p_1 (in this case $v_1(F_2) = v_2(F_2)$). These items that we remove have value lower of b for p_1 and p_2 and we can assign them to p_3 (and p_3 will not take the item that he would have taken in the last iteration in which we had assigned to p_2 the item g), than we assign to F_2 the item g. So now the value of F_2 will be lower or equal to $a - b$.

In all the precedent cases we have removed an item h from F_2 without considering the value for p_3 and so we could think "what if that item had value higher than the ones that already took p_3?" This gives no problem because we already had assumed that p_3 could leave an item like g as last item to p_1 or p_2, so if we give such item to p_2 and than we assign him f as last item this would make no difference to assign h while p_1 was frozen and g as last item, instead if we assign h as last item to p_1, we know that $v_3(F_3) \geq v_3(h)$, so we still have an EFX allocation.

After the problematic assignment After such problematic assignment we have a case in which we can have another problematic assignment, such case is the one in which p_3 has still an item valued a after that p_1 has been unfrozen. If this happens we have that since we start from having all players that are EF towards each other except for p_2 that has $v_2(F_2) + c > v_2(F_1) \geq v_2(F_2)$ we have that we can obtain an EFX allocation by following the rules for such second problematic assignment. Indeed each problematic assignment has been solved by leading to an EF allocation except for the unfrozen player who can still envy the frozen one for at most c, so in this case is enough that if we have to chose one player to freeze between p_1 and p_2, we chose to freeze p_2, avoiding to let him envy for more than c p_1. We can do such a choice since if both $p1$ and $p2$ are involved in another problematic assignment than both have that the remaining items value only c.

Chapter 4

Conclusions

In this thesis, I have shown that by starting from the idea provided in the *Match&Freeze* algorithm [Ama+21] we can obtain an algorithm for obtaining an EFX allocation with two players and three values. Moreover, since the problem of EFX allocation for two players has already been solved with the divide and choose an algorithm, I have improved the approach for two players to deal with the case of three players and three values with the constraint that $c \geq a \mod b$. Before this thesis, the problem for finding EFX allocation for three players had only one result that proved that there is always an EFX allocation for three players. Such proof builds to a pseudo-polynomial algorithm to solve this problem. Instead, I have described an approach to obtain an EFX allocation for three players, three values with a constraint, in polynomial time. Such thesis can be further expanded in two main ways:

- deal with the case in which $c > a \mod b$ does not hold,
- try to expand the work for four players.

While writing this thesis I mainly tried to solve the first of these two points. The main problem for which it is harder dealing with such a case is that for how I built the two players case I have that when we have a problematic assignment and the non frozen players can still take items with value b, they could have to take some items valued c to respect the constraint for which the value of the set of items obtained while the envied player is frozen has to have a value between $a - b - c$ and $a - b$. So in the case of three players, we could have that the two non frozen players could start to envy each other. An approach that I tried and did not work with some rare cases, that maybe could be handled in a different manner, was to delay the non frozen players and let them take value while frozen by avoiding the part with the items valued c while there were still items valued b and then, when the frozen player takes the last item valued b let the other two players take the needed value. Such an approach had problems when one of the two non frozen players finished the items valued b before the frozen player.

Bibliography

[ABM18] Georgios Amanatidis, Georgios Birmpas, and Vangelis Markakis. "Comparing Approximate Relaxations of Envy-Freeness". In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.* Ed. by Jérôme Lang. ijcai.org, 2018, pp. 42–48. DOI: 10.24963/ijcai.2018/6. URL: https://doi.org/10.24963/ijcai.2018/6.

[Ama+21] Georgios Amanatidis et al. "Maximum Nash welfare and other stories about EFX". In: *Theor. Comput. Sci.* 863 (2021), pp. 69–85. DOI: 10.1016/j.tcs.2021.02.020. URL: https://doi.org/10.1016/j.tcs.2021.02.020.

[BK20] Siddharth Barman and Sanath Kumar Krishnamurthy. "Approximation Algorithms for Maximin Fair Division". In: *ACM Trans. Economics and Comput.* 8.1 (2020), 5:1–5:28. DOI: 10.1145/3381525. URL: https://doi.org/10.1145/3381525.

[BL08] Sylvain Bouveret and Jérôme Lang. "Efficiency and Envy-freeness in Fair Division of Indivisible Goods: Logical Representation and Complexity". In: *J. Artif. Intell. Res.* 32 (2008), pp. 525–564. DOI: 10.1613/jair.2467. URL: https://doi.org/10.1613/jair.2467.

[Bud10] Eric Budish. "The combinatorial assignment problem: approximate competitive equilibrium from equal incomes". In: *Proceedings of the Behavioral and Quantitative Game Theory - Conference on Future Directions, BQGT '10, Newport Beach, California, USA, May 14-16, 2010.* Ed. by Moshe Dror and Greys Sosic. ACM, 2010, 74:1. DOI: 10.1145/1807406.1807480. URL: https://doi.org/10.1145/1807406.1807480.

[Car+19] Ioannis Caragiannis et al. "The Unreasonable Fairness of Maximum Nash Welfare". In: *ACM Trans. Economics and Comput.* 7.3 (2019), 12:1–12:32. DOI: 10.1145/3355902. URL: https://doi.org/10.1145/3355902.

[CGH19] Ioannis Caragiannis, Nick Gravin, and Xin Huang. "Envy-Freeness Up to Any Item with High Nash Welfare: The Virtue of Donating Items". In: *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, June 24-28, 2019.* Ed. by Anna Karlin, Nicole Immorlica, and Ramesh Johari. ACM, 2019, pp. 527–545. DOI: 10.1145/3328526.3329574. URL: https://doi.org/10.1145/3328526.3329574.

[CGM20] Bhaskar Ray Chaudhury, Jugal Garg, and Kurt Mehlhorn. "EFX Exists for Three Agents". In: *EC '20: The 21st ACM Conference on Economics and Computation, Virtual Event, Hungary, July 13-17, 2020.* Ed. by Péter Biró et al. ACM, 2020, pp. 1–19. DOI: 10.1145/3391403.3399511. URL: https://doi.org/10.1145/3391403.3399511.

[Fol67] Duncan K. Foley. "Resource allocation and the public sector". In: 1967.

[Gho+18] Mohammad Ghodsi et al. "Fair Allocation of Indivisible Goods: Improvements and Generalizations". In: *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018.* Ed. by Éva Tardos, Edith Elkind, and Rakesh Vohra. ACM, 2018, pp. 539–556. DOI: 10.1145/3219166.3219238. URL: https://doi.org/10.1145/3219166.3219238.

[GMT14] Laurent Gourvès, Jérôme Monnot, and Lydia Tlilane. "Near Fairness in Matroids". In: *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014).* Ed. by Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan. Vol. 263. Frontiers in Artificial Intelligence and Applications. IOS Press, 2014, pp. 393–398. DOI: 10.3233/978-1-61499-419-0-393. URL: https://doi.org/10.3233/978-1-61499-419-0-393.

[Lip+04] Richard J. Lipton et al. "On approximately fair allocations of indivisible goods". In: *Proceedings 5th ACM Conference on Electronic Commerce (EC-2004), New York, NY, USA, May 17-20, 2004.* Ed. by Jack S. Breese, Joan Feigenbaum, and Margo I. Seltzer. ACM, 2004, pp. 125–131. DOI: 10.1145/988772.988792. URL: https://doi.org/10.1145/988772.988792.

[PR20a] Benjamin Plaut and Tim Roughgarden. "Almost Envy-Freeness with General Valuations". In: *SIAM J. Discret. Math.* 34.2 (2020), pp. 1039–1068. DOI: 10.1137/19M124397X. URL: https://doi.org/10.1137/19M124397X.

[PR20b] Benjamin Plaut and Tim Roughgarden. "Almost Envy-Freeness with General Valuations". In: *SIAM J. Discret. Math.* 34.2 (2020), pp. 1039–1068. DOI: 10.1137/19M124397X. URL: https://doi.org/10.1137/19M124397X.

[Var74] Hal R Varian. "Equity, envy, and efficiency". In: *Journal of Economic Theory* 9.1 (1974), pp. 63–91. ISSN: 0022-0531. DOI: https://doi.org/10.1016/0022-0531(74)90075-1. URL: https://www.sciencedirect.com/science/article/pii/0022053174900751.

[Wik21] Wikipedia contributors. *Fair division — Wikipedia, The Free Encyclopedia.* [Online; accessed 2-December-2021]. 2021. URL: https://en.wikipedia.org/w/index.php?title=Fair_division&oldid=1041463155.