Fair division of mixed and indivisible items

Federico Fattorini

July 28, 2023

Contents

1 Introduction

The problem of fair division, referred as FD, is about finding the best way to allocate a certain bundle of items to a number of participants following their subjective preferences. The bundle to be divided is defined as the *manna*, and can be composed of various kinds of item, depending on the particular problem we are dealing with. In particular, it can be composed of a set of desiderable *goods*, as in the original formulation of the problem, or of a set of bads, also referred as *chores*, with reference to the direct application of assigning tasks to different people; the manna could also be *mixed*, i.e. a composition of goods and chores, and the distinction among them could be subjective to the agents preferences. One straightforward example of this mixed scenario is the compensation of an agent with some goods for doing some chores.
Finally, the manna could be composed of divisible objects (e.g., a cake), or indivisible ones; the formalisation of the latter case will be the main topic of this work, since some properties of the former should be adapted or could be relaxed in the latter case.
The application of FD are many, starting from the juridical field, as in the subdivision of heirlooms among siblings or of the assets of a dissolving partnership (e.g., marriage, bankruptcy of a firm), to end with the social task of assigning responsibilities to a set of workers (family chores, teaching loads, contracts, etc.).
The aim of this work is then to focus on some of the properties required by a fair division in the divisible case, and extend them to the indivisible scenario, concentrating on how to find solutions that satisfy them by the use of particular algorithms.

1.1 Basic assumptions

In the following description, we will deal mostly with the case in which participants have all equal rights (or responsibilities) to the manna; thus, the division rules will stick just to the preferences of the participants, and in particular, to the utilities they will receive from different bundles of the manna. For this reason, it is necessary to make 3 assumptions regarding the preferences of agents:

- Agents are selfish, meaning that they care only about their particular final share of the manna, without caring about the satisfaction the others will get.
- The full individual responsibility for the particular preference is on the agents; as long as their preferences are respected, they don't complain about the lowness of their shares.
- Agents do not assign values to their utilities in such a way that manipulates the division rules.

2 EF and FS: definitions in the cake-cutting framework

In order to find a fair division rule, we first need to define what does it mean to be fair. Historically, there are two properties that can be used as a definition of fairness, relying on different ideological qualities of a division: *fair share* property and *envy-freeness*. In order to understand them, we illustrate two examples in the case of divisible manna.
Consider the case in which n agents need to divide the cake C, a measurable set in Euclidean space; let $u_i(\pi_i)$ $(i = 1, 2...n)$ be the utility for the agent i for consuming the share π_i, where π_i is a measurable subset of C and u_i is an atomless non-negative measure. As we will do for the rest of the work, we focus on the case of *additive* utility, i.e., $u_i(\pi_i \cup \pi_i') = u_i(\pi_i) + u_i(\pi_i')$.
The first property that can be taken into account is the *fair share* one:

Definition 1. A measurable partition π of C satisfies the *fair share* property (FS) if:

$$u_i(\pi_i) \geq \frac{1}{n} u_i(C) \quad \forall i = 1, 2...n$$

Instead, considering the fact that agent i *envies* agent j if $u_i(\pi_i) < u_i(\pi_j)$ (i.e., agent i prefers agent j's bundle to his one), we can hypothesize that a criterion for a partition to be fair is that no agent prefers the share of anyone of the others. This is the rationale behind the concept of *envy-freeness*. Formally:

Definition 2. A measurable partition π is *envy-free* (EF) if:

$$u_i(\pi_i) \geq u_i(\pi_j) \quad \forall i, j = 1, 2...n$$

By additivity , just summing up the EF inequalities for all the agents j, it is straightforward to see that EF implies FS:

$$\sum_{j \in N} u_i(\pi_i) = n u_i(\pi_i) \geq \sum_{j \in N} u_i(\pi_j) = u_i(C) \quad \forall i \in N$$

The vice versa is not true.
The first historical example of an EF rule can be found in the game "Divide and Choose", which is first mentioned in the Bible, in the Book of Genesis.
The game is played by two agents: player 1 decides how to partition a divisible manna, player 2 decides which partition to claim. If the divider splits the manna in two equal parts, then he cannot envy the chooser, because the latter cannot pick a better share; the same holds for the chooser, because he can pick a share that is at least as good as the divider's one. Following this rule, no player prefers the other one's share, so the partition is EF.
To see an example for the FS property, we need to get back to the cake-cutting problem with n agents described above. A division rule that satisfies FS was found by Steinhaus in 1948 and it is the following:

1. Agent 1 cuts a share z_1 that is fair for him according to his utility u_1, such that $u_1(z_1) = \frac{1}{n} u_1(C)$. Then, he offers z_1 to agent 2.

2. Agent 2 has two possibilities: if $u_2(z_1) \geq \frac{1}{n} u_2(C)$, then he cuts a smaller share $z_1' \subset z_1$ such that $u_2(z_1') = \frac{1}{n} u_2(C)$ and offers it to agent 3; otherwise he passes z_1 to agent 3.

3. The round repeats for all the agents; the last agent to cut gets the share and is out.

4. The rule then repeats with the remaining cake and n-1 agents.

If each agent makes his move truthfully, all of them are guaranteed to get a share with utility worth at least $\frac{1}{n} u_i(C)$.
Before moving to the indivisible case, it is important to highlight the fact that both EF and FS properties are subjective to the particular result the division rule aims to accomplish. There is no reason, a priori, to state that one of them is the best fairness test for a division rule and, of course, there is no need to require both of them. Moreover, many other concepts of fairness have been formalised in the fair division literature; the choice of dealing mainly with FS and EF in this paper is guided by the straightforward relaxations and applications of these concepts, as we will see in the next sections.

3 Extension to the indivisible case

3.1 Definition of the problem

Now we want to modify the properties above to apply them in a different scenario, and see how some requirements should be relaxed to achieve a feasible solution. The case we are going to deal with is the one of a fair division problem with indivisible items, which may have either positive or negative utility depending on each agent's preference. Indivisible items means that each object could either be totally in one agent's bundle or not: there cannot be a situation in which two agents or more share a part of it (with the exception of fractional allocations, that we will introduce later).
Formalising the problem, we define an *instance* as $I = (N, O, U)$, where:

- $N = \{1, 2..., n\}$ is the set of agents.
- $O = \{o_1, o_2, ..., o_m\}$ is the set of the indivisible items in the manna (each subset is a *bundle* of items).
- U is a n-tuple of utility functions $u_i : 2^O \longrightarrow \mathbb{R}$.

In the following we will use the notation $u_i(o) = u_i(\{o\})$. As anticipated above, an item o can be a *good*, a *chore* or a *null item* for agent i if, respectively, $u_i(o) > 0$, $u_i(o) < 0$ or $u_i = 0$. Noting that an item can be a good, a chore or a null item according to the personal preference of each agent, the following discrimination is necessary:

- An item is a *subjective good* if $u_i(o) > 0$ for some agent i in N and $u_j(o) \leq 0$ for some other agent j in N (for *subjective chores*, $u_i(o) < 0$ and $u_j(o) \geq 0$ for some i, j in N).
- An item is an *objective good* if $u_i(o) > 0 \quad \forall i \in N$ (for *objective chores*, $u_i(o) < 0 \quad \forall i \in N$).

Note that in this distinction an item could be a subjective good and chore at the same time, but an objective good cannot be a null item or a chore.
As we did in the previous section, we assume *additivity* for the utility function, i.e., $u_i(X) = \sum_{o \in X} u_i(o)$ for each bundle $X \subseteq O$. This implies $u_i(\emptyset) = 0 \quad \forall i \in N$, since $u_i(\emptyset) = u_i(\emptyset) + u_i(\emptyset)$.
We say that agent i *weakly prefers* item o to item o' if $u_i(o) \geq u_i(o')$, and that i *strictly prefers* o to o' if $u_i(o) > u_i(o')$ must hold.
We define an *allocation* π as a function $\pi : N \longrightarrow 2^O$ such that $\cup_{i \in N} \pi(i) = O$ and $\pi(i) \cap \pi(j) = \emptyset$ for every agent $i, j \in N$.

3.2 Different requirements for a fair allocation

In this subsection we will try to extend the FS and EF properties to the indivisible case, and we will see how these properties are not always useful for a manna made of indivisible items.
As said above, given a certain allocation π and two agents i and j, we say that i *envies* j if $u_i(\pi(i)) < u_i(\pi(j))$. An allocation is said to be *envy-free* (EF) if no agent envies any other agent. We can even define envies restricted to particular bundles:

given an allocation π and a bundle X, agent i *envies agent j with respect to X* if $u_i(\pi(i) \cap X) < u_i(\pi(j) \cap X)$.
A *FS allocation* guarantees to each agent i a bundle of items $\pi(i)$ that is at least his fair share $\frac{1}{n}u_i(O)$. Even in the case of mixed manna, EF implies FS, as long as utilities satisfy additivity. The proof is the same illustrated above.
As anticipated, these properties are not useful anymore in the case of indivisible manna, for the simple reason that they are not feasible in most of the problems. The simplest example of such failure is the case of just one good and two agents, for obvious reason. Hence, the only possibility to deal with indivisible manna is the relaxation of these properties.
First, we can modify EF considering that the envy of an agent can disappear if one good is removed from the envied agent's bundle, or a chore is removed from the unhappy player's bundle. Formally:

Definition 3. Given an allocation π, we say that *i envies j by more than one item* if i envies j and $u_i(\pi(i) \setminus \{o\}) < u_i(\pi(j) \setminus \{o\}) \quad \forall o \in \pi(i) \cup \pi(j)$.

Definition 4. An allocation π is *envy-free up to one item* (EF1) if $\forall i, j \in N$ i does not envy j by more than one item.

It is clear that EF implies EF1.
The relaxation of FS follows the same reasoning behind EF1: each agent should get his fair share receiving one additional good or removing one of the chores in his bundle.

Definition 5. An allocation π satisfies *fair share up to one item* (FS1) if for each agent $i \in N$ one of the following is true:

- $u_i(\pi(i)) \geq \frac{1}{n}u_i(O)$ (FS)
- $u_i(\pi(i))$+ $u_i(o) \geq \frac{1}{n}u_i(O)$ for some $o \in O \setminus \pi(i)$
- $u_i(\pi(i))$- $u_i(o) \geq \frac{1}{n}u_i(O)$ for some $o \in \pi(i)$

The relation between EF1 and FS1 is the same of EF and FS: the former implies the latter. This is stated in the following proposition:

Proposition 1. *For additive utilities, an allocation EF1 satisfies FS1*

Proof. Proof: Suppose $|N| \geq 2$ and $O \neq \emptyset$; the case with $|N| \leq 1$ or $O = \emptyset$ is trivial. Let π be a EF1 allocation. We have three cases for each agent i, depending on the possible bundle he receives:

- $\pi(i) = O$ for some agent i. If $u_i(O) \geq 0$, then $u_i(\pi(i)) = u_i(O) \geq \frac{1}{n}u_i(O)$, and agent i obviously gets his fair share. If $u_i(O) < 0$, we focus on any other player $j \neq i$. Allocation π is EF1 by hypothesis, so $u_i(\pi(i)) - u_i(o) \geq u_i(\pi(j)) = u_i(\emptyset) = 0$ for some $o \in \pi(i)$. Hence $u_i(\pi(i)) - u_i(o) \geq 0 > \frac{1}{n}u_i(O)$ for some $o \in \pi(i)$.

- $\pi(i) = \emptyset$ for some agent i. If $u_i(O) \leq 0$, then $u_i(\pi(i)) = 0 \geq \frac{1}{n}u_i(O)$. Now, consider $u_i(O) > 0$: since π is EF1, given any agent $j \neq i$, $u_i(\pi(i)) \geq u_i(\pi(j))$, or $u_i(\pi(i)) \geq u_i(\pi(j) \setminus o)$ for some $o \in \pi(i)$; hence $u_i(\pi(i)) = u_i(\emptyset) = 0 \geq u_i(\pi(j) - u_i(o)$. If we consider $o^* \in argmax_{o \in O} u_i(o)$, $u_i(\pi(i)) + u_i(o^*) \geq u_i(\pi(j)) \quad \forall j \in N$, which implies $u_i(\pi(i)) + u_i(o^*) \geq \frac{1}{n}u_i(O)$ (summing on all the players j).

- $O \setminus \pi(i) \neq \emptyset$ and $\pi(i) \neq \emptyset$. Let $x = \max_{o \in O \setminus \pi(i)} u_i(o)$ and $y = \min_{o \in \pi(i)} u_i(o)$ (respectively, the utility of the best good not in $\pi(i)$ and of the worst chore in $\pi(i)$). Because of EF1, for any agent $j \neq i$ one of the following must be true:

$$u_i(\pi(i)) \geq u_i(\pi(j))$$

$$u_i(\pi(i)) + x \geq u_i(\pi(j))$$
$$u_i(\pi(i)) - y \geq u_i(\pi(j))$$

Defining $b^*+ = \max\{x, -y, 0\}$, these relations are equivalent to:

$$u_i(\pi(i)) + b^* \geq u_i(\pi(j)) \quad \forall j \neq i$$

Finally, summing the previous inequality on all players j:

$$u_i(\pi(i)) + b^* \geq \frac{1}{n} \sum_{j \in N} u_i(\pi(j)) = \frac{1}{n} u_i(O)$$

Therefore, allocation π satisfies FS1.

□

3.3 Efficient allocations

In addition to the different fairness concepts introduced above, we can define a criterion to establish if an allocation is efficient or not. The most common one is *Pareto optimality*, which essentially describes the case in which it is impossible to improve the utility of an agent without diminishing any other one.
To formalise the idea, we first need to introduce the concept of *Pareto improvement* in our framework:

Definition 6. Given an allocation π, another allocation π' is a *Pareto improvement* of π if $u_i(\pi') \geq u_i(\pi) \quad \forall i \in N$ and $u_j(\pi') > u_j(\pi)$ for some $j \in N$.

Definition 7. An allocation π is *Pareto optimal* (PO) if there is no other allocation that is a Pareto improvement of π.

With the definition of Pareto optimality, we have completed the background necessary to define which qualities we could require from a fair division. In the following example, we check for the defined properties in a particular problem.

Example 1. Consider an instance with $n = 4$ agents and the indivisible manna $O = \{o_1, o_2, ..., o_9\}$. The utilities for each agent are expressed in the following table:

	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9
Agent 1	1	-1	2	1	-2	-4	-6	-1	-1
Agent 2	4	-3	6	2	-2	-2	-2	-1	-1
Agent 3	0	11	8	11	0	0	0	10	0
Agent 4	0	11	8	11	0	0	0	0	10

We consider the allocation π:

$$\pi(1) = \{o_2, o_4\}, \quad \pi(2) = \{o_1, o_3, o_5, o_6, o_7\}, \quad \pi(3) = \{o_8\}, \quad \pi(4) = \{o_9\}$$

Looking at the table of utilities, by additivity, we get:

$$u_1(\pi(1)) = -1+1 = 0; \quad u_2(\pi(2)) = 4+6-2-2-2 = 4; \quad u_3(\pi(3)) = 10; \quad u_4(\pi(4)) = 10$$

Allocation π satisfies FS, since:

$$u_1(\pi(1)) = 0 \geq -\frac{11}{4} = \frac{1}{n}u_1(O), \quad u_2(\pi(2)) = 4 \geq \frac{1}{4} = \frac{1}{n}u_2(O),$$

$$u_3(\pi(3)) = 10 \geq 10 = \frac{1}{n}u_3(O), \quad u_4(\pi(4)) = 10 \geq 10 = \frac{1}{n}u_4(O)$$

Allocation π does not satisfy neither EF nor EF1, because both agents 3 and 4 are envious by more than one item of agent 1:

$$u_3(\pi(3)) = 10 < u_3(\pi(1)) = 22, \quad u_4(\pi(4)) = 10 < u_4(\pi(1)) = 22$$

$$u_3(\pi(3)) = 10 < u_3(\pi(1) \setminus o_2) = 11, \quad u_3(\pi(3)) = 10 < u_4(\pi(1) \setminus o_4) = 11$$
$$u_4(\pi(4)) = 10 < u_4(\pi(1) \setminus o_2) = 11, \quad u_4(\pi(4)) = 10 < u_4(\pi(1) \setminus o_4) = 11$$

Finally allocation π is not Pareto optimal, since we can have a Pareto improvement by moving the items o_5, o_6, o_7 from agent 1 and 2 to agent 3 or 4; the former ones receive negative utilities by these items, while the latter get no damage from them.

4 Round-robin rule failure and extension to find EF1 allocations

In the case of a manna composed by all goods or chores, an EF1 allocation can be found in polynomial time by an algorithm known as *round-robin rule*. The algorithm is very simple: in an order established before the start of the algorithm, each agent, in his round, picks the most preferred item from the remaining ones in the manna, until no item is left to pick. However, in the more general case of a mixed manna, such an algorithm fails to find an EF1 allocation. This is proven by a simple counterexample:

Proposition 2. *The round-robin rule does not satisfy EF1*

Proof. Proof: Consider the instance with 2 agent and 4 items as a manna, with the utilities described by the following table:

	o_1	o_2	o_3	o_4
agent 1	2	-3	-3	-3
agent 2	2	-3	-3	-3

Agent 1 chooses first: he picks the only good, then there is no difference among the remaining chores for the other agents; hence, the final allocation π is (o_2, o_3, o_4 can be exchanged):

$$\pi(1) = \{o_1, o_2\}, \quad \pi(2) = \{o_3, o_4\}$$

Obviously agent 2 envies agent 1. Moreover, even if we remove the good from agent 1, the envy persists. Hence, the allocation found by the round-robin rule is not EF1. □

Even if the original algorithm is not able to find an EF1 allocation, the idea could be used to implement a new version which successfully achieve it. This is called the *double round-robin algorithm*, for the simple reason that the original algorithm is essentially applied twice, in normal and reversed order. First, in a clockwise order each agent picks an item with non-positive utility for him, until no items of such kind are left; then, in the opposite order, each agent picks items with positive utility for him. To be more specific, given an instance $I = (N, O, U)$, the algorithm works as follow:

Algorithm 1 Generalised round-robin algorithm

```
1: $\pi(i) = \emptyset \quad \forall i \in N$
2: Partition $O$ in $O^+ = \{o \in O : \exists i \in N \quad \text{s.t.} \quad u_i(o) > 0\}$ and $O^- = \{o \in O : \forall i \in N \quad u_i(o) \leq 0\}$. Suppose $|O^-| = an - k$ with $a \in \mathbb{N}$ and $k \in \{0, ..., n-1\}$
3: Add k objective (dummy) null items that were not in the original manna to $O^-$ in order to have $|O^-| = an$
4: Agents pick their most preferred items (one for round) in $O^-$ according to the round-robin rule, in the order $(1, 2, ..., n)$, until $O^-$ is completely allocated
5: An inverse round-robin rule with order $(n, n-1, ..., 1)$ allocates all the items in $O^+$ (each agent picks the most preferred item in his round). If there are no items with strictly positive utility for a particular agent, he picks a dummy null item that was not in the original manna
6: Removes the dummy items from the allocation
7: **return** the resultant allocation $\pi$
```

The result of the algorithm is an EF1 allocation, as it is stated by the following theorem, which also gives an esteem for the running time (with n players and m items to allocate).

Theorem 1. *For additive utilities, the double round-robin algorithm returns an EF1 allocation in* $O(\max\{m \log m, mn\})$ *time*

Proof. First we note that adding the k dummy items guarantees that all agents have the same number of chores or null items. Now, consider the resultant allocation π and two agents i and j, with $i < j$. We define $c_{i,t}$ and $c_{j,t}$ as the items respectively allocated to agent i and j at round t in the first round-robin sequence (at step 4), where $t = 1, 2, ..., a$; instead, we denote $g_{i,t}$ and $g_{j,t}$ as the items respectively allocated to agent i and j at round t in the inverted round-robin sequence (at step 5), where now $t = 1, 2, ..., b$.
First, noting that the t-th item $c_{i,t}$ in O^- allocated to i is weakly preferred by agent i to the t-th item $c_{j,t}$ in O^- allocated to agent j, we conclude that agent i does not envy agent j with respect to O^-. Formally:

$$u_i(\pi(i) \cap O^-) = \sum_{t=1}^{a} u_i(c_{i,t}) \geq \sum_{t=1}^{a} u_i(c_{j,t}) = u_i(\pi(j) \cap O^-)$$

Considering now O^+, for each item $g_{j,t}$ picked by agent j at time $t = 2, 3, ..., b$, agent i can pick an item $g_{i,t-1}$ that is weakly preferred by i to $g_{j,t}$, before j's turn. Therefore, the only item which agent i could envy to agent j is $g_{j,1}$: if we remove it from j's bundle, the envy disappears. Namely:

$$u_i(\pi(i) \cap O^+) = \sum_{t=1}^{b} u_i(g_{i,t}) \geq \sum_{t=2}^{b} u_i(g_{j,t}) = u_i(\pi(j) \cap O^+) \setminus \{g_{1,j}\})$$

Combining the two relations, we see that agent i does not envy agent j by more than one item:

$$u_i(\pi(i)) \geq u_i(\pi(j) \setminus \{g_{j,1}\})$$

Now, we consider agent j's envy for agent i; for the same reason above, there is no envy from agent i to agent j with respect to O^+, since the former picks before the latter in the inverted round-robin rule. Hence:

$$u_j(\pi(j) \cap O^+) = \sum_{t=1}^{b} u_j(g_{j,t}) \geq \sum_{t=1}^{b} u_j(g_{i,t}) = u_j(\pi(i) \cap O^+)$$

Then, for each item $c_{i,t} \in O^-$ picked by i in the round robin round $t = 2, 3, ..., a$, agent j could have picked in the round before an item $c_{j,t-1}$ which he weakly prefers to $c_{i,t}$. The only possible reason of envy is the last object $c_{j,a}$. Formally, remembering that $c_{i,1}$ is a bad :

$$u_j((\pi(j) \cap O^-) \setminus \{c_{j,a}\}) = \sum_{t=1}^{a-1} u_j(c_{j,t}) \geq \sum_{t=2}^{a} u_j(c_{i,t}) \geq \sum_{t=1}^{a} u_j(c_{i,t}) = u_j(\pi(i))$$

Again, the two previous relations lead to the conclusion that:

$$u_j(\pi(j) \setminus \{c_a^j\} \geq u_j(\pi(i))$$

Finally, we conclude that in both cases no agent envies another one by more than one item: once the dummy items are removed, the resultant allocations is EF1, since removing an objective null item does not affect the utilities of the agents.
Now, we focus on the running time. Partitioning the manna in O^+ and O^- requires mn iterations, since the values of the utility for m items must be computed for n agents. The two applications of the round-robin rule require $\mathrm{O}(m \log m)$ time: the objects can be sorted according to the preferences of each agents at the beginning (to find the most preferred items), and the 2 picking sequences require at most m iterations. Putting all together, the total running time is $O(\max\{m \log m, mn\})$.

□

5 Generalised Adjusted Winner: PO and EF1 allocations

In the previous section we totally neglected the need of efficiency in finding an EF1 allocation. The reason is that finding an EF1 and PO allocation is an unresolved problem in general, even if we restrict the analysis to a manna made of just chores. The first approach one could think of is to find EF1 and look for Pareto improvements; however, this is a NP-hard problem, and Pareto improvements may not preserve EF1. So, even neglecting computation complexity, improving efficiency could break EF1, as shown in the following example:

Example 2. Consider an instance with 3 agents and 4 items, with the following utilities:

	o_1	o_2	o_3	o_4
Agent 1	1	3	1	2
Agent 2	3	2	1	0
Agent 3	4	3	1	2

A possible EF1 allocation π is the following:

$$\pi(1) = \{o_3\}, \quad \pi(2) = \{o_1, o_4\}, \quad \pi(3) = \{o_2\}$$

We can find a Pareto improvement by moving o_4, which gives 0 utility to agent 2, to agent 3, who considers it as a good:

$$\pi'(1) = \{o_3\}, \quad \pi'(2) = \{o_1\}, \quad \pi'(3) = \{o_2, o_4\}$$

However, the new allocation π' is not EF1 anymore, since agent 1 now envies agent 3 by more then one item.

Hence, a general solution to the problem with n agents is still object of research, but we can find an answer to the problem with two agents, generalising an algorithm known as *Adjusted Winner*, which finds an EF allocation for two agents in the case of good manna. The original algorithm is the following:

1. Agent 1 is called the *winner*, agent 2 the *loser*.
2. For every item o in the manna, if $u_1(o) > u_2(o)$, o is given to the winner, otherwise it is given to the loser.
3. The items given to the winner are ordered increasingly by the ratio $\frac{u_1(o)}{u_2(o)}$
4. Fraction of items in the winner allocation are continuously given to the the loser, in the sequence specified above (from the smallest to the largest), until an allocation which gives equal utility to both agents is obtained

The result of the algorithm is a *fractional allocation* (one item could be split between the two agents), which is EF and PO. However, as anticipated above, it works only with positive utility items.
The idea could be extended to deal with mixed manna and to find an allocation that does not split any item. This is implemented in the *Generalised Adjusted Winner* algorithm, which essentially can be described by the following steps, that will be formalised later:

1. All subjective chores are given to the agent which considers them as goods or null items. Analogously, all subjective goods are given to the agent which consider them as goods. Hence, only objective items are left.
2. One agent is chosen to be the *winner* w, the other is the *loser* l
3. All goods are given to the winner, all chores to the loser.
4. The items are sorted by $\frac{|u_l(o)|}{|u_w(o)|}$ in decreasing order.
5. Items are reallocated with the following rule, respecting the previous order: if o is a good, it is moved from the winner to the loser; if it is a chore, it is moved from the loser to the winner. The reallocation occurs one item at the time and stops only when the loser does not envy the winner by more then one item.

Note that the previous rule left out the case of objective null items; of course, we can allocate them arbitrarily, since they do not change the utilities of both agents.
To better understand how the algorithm works, consider the following example:

Example 3. We study an instance with two agents, 7 items to allocate and the additive utilities in the following table, which also reports the ratio $\frac{|u_l(o)|}{|u_w(o)|}$ for each object:

	o_1	o_2	o_3	o_4	o_5	o_6	o_7
agent 1	1	-1	2	1	-2	-4	-6
agent 2	4	-3	6	2	-2	-2	-2
$\frac{\lvert u_l(o)\rvert}{\lvert u_w(o)\rvert}$	4	3	3	2	1	$\frac{1}{2}$	$\frac{1}{3}$

Consider agent 1 as the winner. First, the algorithm allocates all the goods to the winner, and the chores to agent 2. Then o_1 is moved from agent 1 to agent 2, and o_2 from agent 2 to agent 1. The algorithm ends with the third object moving to agent 2, since the loser stop being envious by more than one item. Therefore, the resultant EF1 and Pareto optimal allocation is:

$$\pi(1) = \{o_2, o_4\}, \quad \pi(2) = \{o_1, o_3, o_5, o_6, o_7\}$$

Now, we need to formalise all the concept introduced in this section. We start with the theorem guaranteed by the Generalised Adjusted Winner algorithm:

Theorem 2. *For two agents with additive utilities, a PO and EF1 allocation always exists and can be computed in $O(m^2)$ time.*

Before proving the theorem, we need to rewrite the algorithm in a more precise way:

Algorithm 2 Generalised Adjusted Winner

```
1: $\pi(i) = \emptyset \quad \forall i \in \{w, l\}$
2: Let $O^*_w = \{o \in O : u_w(o) \geq 0 \quad \text{and} \quad u_l(o) \leq 0\}$ and $O^*_l = \{o \in O : u_l(o) \geq 0 \quad \text{and} \quad u_w(o) < 0\}$
3: Let $O^+ = \{o \in O : u_i(o) > 0, \quad i \in \{w, l\}\}$ and $O^- = \{o \in O : u_i(o) < 0, \quad i \in \{w, l\}\}$
4: Add each item $o \in O^+ \cup O^*_w$ to $\pi(w)$, and each item $o \in O^- \cup O^*_l$ to $\pi(l)$
5: Sort items in $O^+ \cup O^-$ in decreasing order of $\frac{|u_l(o)|}{|u_w(o)|}$. We denote the ordered items with $o_1, ..., o_k$ ($k = |O^+ \cup O^-|$)
6: set t=1
7: **while** agent $l$ envies agent $w$ by more then one item, **do**:
8:     **if** $o_t \in O^+$, **then**
9:         $\pi(w) = \pi(w) \setminus \{o_t\}$ and $\pi(l) = \pi(l) \cup \{o_t\}$
10:    **else if** $o_t \in O^-$, **then**
11:        $\pi(w) = \pi(w) \cup \{o_t\}$ and $\pi(l) = \pi(l) \setminus \{o_t\}$
12:    t=t+1
13: **return** $\pi$
```

Now, we state the following lemma, which will be used in the proof of the theorem above and which gives a more precise indication of how the algorithm behaviour is efficient.

Lemma 3. *In the Generalised Adjusted Winner, the allocation π is Pareto optimal during each step of reallocation (i.e., from step 7 to the end)*

Proof. The allocation π resulting by step 4 of the algorithm is Pareto optimal by construction.
Now, consider a generic step of the while in line 7, and suppose by absurd that there is an allocation π' which is a Pareto improvement of π. We assume that no items are moved from O^*_w or O^*_l in π', since such change would be obviously inefficient.
Consider then each good $o \in O^+$ and chore $o \in O^-$, we define for each $i, j \in \{w, l\}$ with $i \neq j$ the following sets:

- G_{ii} is the set of goods in $\pi(i) \cap \pi'(i)$
- C_{ii} is the set of chores in $\pi(i) \cap \pi'(i)$
- G_{ij} is the set of goods in $\pi(i) \cap \pi'(j)$
- C_{ij} is the set of chores in $\pi(i) \cap \pi'(j)$

Consider the case with $u_l(\pi'(l)) > u_l(\pi(l))$ and $u_w(\pi'(w)) \geq u_w(\pi(w))$. By additivity, and considering that by definition G_{ww}, G_{ll}, C_{ww} and C_{ll} are allocated to the same agent in π and π', the inequalities can be rewritten as:

$$u_l(G_{wl}) + u_l(C_{wl}) - u_l(G_{lw}) - u_l(C_{lw}) > 0$$

$$u_w(G_{lw}) + u_w(C_{lw}) - u_w(G_{wl}) - u_w(C_{wl}) \geq 0$$

Now, remembering that all the goods are initially assigned to the winner and the chores to the loser, we note that the items in G_{lw} and in C_{wl} are those reassigned by the while loop, while items in G_{wl} and in C_{lw} are assigned by initial allocation. Therefore, items in G_{lw} and in C_{wl} are moved before items in G_{wl} and in C_{lw}. Considering the fact that the order is established according to $\frac{|u_l(o)|}{|u_w(o)|}$ in a decreasing manner, we can define an $\alpha \geq 0$ such that:

$$\max_{o \in G_{wl} \cup C_{lw}} \frac{|u_l(o)|}{|u_w(o)|} \leq \alpha \leq \min_{o \in G_{lw} \cup C_{wl}} \frac{|u_l(o)|}{|u_w(o)|}$$

This is equivalent to say that:

$$u_l(G_{wl}) \leq \alpha u_w(G_{wl}), \quad u_l(G_{lw}) \geq \alpha u_w(G_{lw}),$$

$$-u_l(C_{wl}) \geq -\alpha u_w(C_{wl}), \quad -u_l(C_{lw}) \leq -\alpha u_w(C_{lw})$$

Putting these relation together with the inequalities for the Pareto improvement, and using that $\alpha \geq 0$ we get:

$$0 < u_l(G_{wl}) + u_l(C_{wl}) - u_l(G_{lw}) - u_l(C_{lw}) \leq$$

$$\leq -\alpha[u_w(G_{lw}) + u_w(C_{lw}) - u_w(G_{wl}) - u_w(C_{wl})] \leq 0$$

which is obviously a contradiction.
The same argument applies when $u_l(\pi'(l)) \geq u_l(\pi(l))$ and $u_w(\pi'(w)) > u_w(\pi(w))$.

□

Now, we can finally prove that the Generalised Adjusted Winner finds a PO and EF1 allocation.

Proof. The PO property is guaranteed by the previous lemma, so it remains to prove that the resulting allocation satisfies EF1.
First, note that by the previous lemma it is impossible to achieve a situation where both agents envy each other: in that case, exchanging the resultant bundles we would obtain a Pareto improvement, in contradiction to the lemma.
Suppose in the allocation π the loser envies the winner; hence, the winner cannot envy the loser. Therefore, π is EF1, because by the stopping condition of the algorithm the loser cannot envy the winner by more then one item at the end of the execution.
Consider now when the winner envies the loser (the loser cannot envy the winner in this case). We define as π' the allocation before the last transfer in the while loop. Let $W = \pi'(w) \cap \pi(w)$ and $L = \pi'(l) \cap \pi(l)$, which are respectively the items in the winner and loser bundles excluding the last transferred item. By construction, the loser still envies the winner by more than one item at π', which implies $u_l(L) < u_l(W)$. Suppose now, by absurd, that the winner envies the loser by more than one item at π, which implies $u_w(W) < u_w(L)$. Consider the last item o moved by the while loop:

- if o is a good moved from the winner to the loser, allocating W to the loser and $L \cup \{o\}$ to the winner would be a Pareto improvement of π', contradicting the previous lemma

- if o is a chore moved from the loser to the winner, allocating $W \cup \{o\}$ to the loser and L to the winner would be a Pareto improvement, a contradiction for the same lemma.

Therefore, the winner does not envy the loser by more than one item at π: allocation π is EF1.
Focusing on the running time, first the items can be sorted in $O(m \log m)$ time; then, the reallocation process takes $O(m^2)$ time, because at the beginning of m iterations the stopping condition requires m comparisons to check if the loser envies the winner by more than one item. Therefore, the total running time is $O(m^2)$.

□

6 Finding a PO and FS1 allocation

As stated above, finding an EF1 allocation satisfying Pareto optimality is an unresolved problem for a number of agents greater than 2. However, if we change the EF1 property to FS1, it is possible to find a suitable division for any number of agents. In this section, we will show an algorithm to find such allocation, and we will see how the result of such procedure satisfies a stronger property, called FPO.

6.1 Fractional allocations and FPO property

In the first section, we stated that in the usual framework agents have all equal rights to the manna. In this one, the results apply to the more general case of *weighted agents*, which means that for each agent i we can assign a weight $b_i > 0$ such that $\sum_{i \in N} b_i = 1$; we can imagine the weight as some sort of right on a particular claim on the manna by the agents, whose division is no more based only on the preferences of the agents. Note that the case $b_i = \frac{1}{n}$ is the one studied in the previous sections.
Moreover, up to now we considered only *integral* allocation, which means that each item is allocated to a single agent. However, we can define a *fractional* allocation by $x = (x_1, x_2, ..., x_n)$, where x_i is the allocation of agent i and $x_{i,o}$ is the fraction of item o allocated to agent i (clearly $\sum_{i \in N} x_{i,o} = 1$). In this setting, the utility that an agent i gets from a particular allocation is $u_i(x_i) = \sum_{o \in O} u_i(o) x_{i,o}$, considering again the unweighted case.
We can now extend the definition of FS property to the weighted case with fractional allocation:

Definition 8. An allocation x satisfies *weighted fair share* if for each agent $i \in N$:

$$u_i(x_i) \geq b_i u_i(O)$$

Note that a fractional allocation which gives a share b_i of each item to each agent i satisfies weighted FS. To see this, consider the allocation x such that, for each agent i, $x_{i,o} = b_i \quad \forall o \in O$; hence:

$$u_i(x_i) = \sum_{o \in O} x_{i,o} u_i(o) = \sum_{o \in O} b_i u_i(o) = b_i u_i(O)$$

Therefore, the fractional allocation x satisfies weighted FS. We will use this result in the next section.

We can also define an FS1 property in the case of integral allocation with weight:

Definition 9. An integral allocation x is *fair share up to one item* (FS1) if for each agent $i \in N$ one of the following is true:

- $u_i(\pi(i)) \geq b_i u_i(O)$ (FS)
- $u_i(\pi(i))$+ $u_i(o) \geq b_i u_i(O)$ for some $o \in O \setminus \pi(i)$
- $u_i(\pi(i))$- $u_i(o) \geq b_i u_i(O)$ for some $o \in \pi(i)$

Finally, with the notion of fractional allocation, we can introduce a new efficiency concept, *fractional Pareto optimality*:

Definition 10. An allocation satisfies *fractional Pareto optimality* (FPO) if it cannot be improved by any fractional allocation.

Since integral allocations are a subset of fractional allocation, it is clear that FPO implies PO.

6.2 Pareto improvements and FS1/FS property

In the next subsection we will describe an algorithm that finds an allocation satisfying FS1 and PO. However, such an algorithm starts with a FS fractional allocation, and looks for Pareto improvements. As for EF1 allocations, Pareto improvements do not necessary preserve the FS1 properties, as we can see from the following example:

Example 4. Consider an instance with 3 agents and 31 items: one single item o_A, and two sets of items B and C, composed respectively by 10 and 20 smaller and identical items. We consider the following utilities:

	o_A	$B = \{o_{b,1}, ..., o_{b,10}\}$	$C = \{o_{c,1}, ..., o_{c,20}\}$
agent 1	0.3	0.2	0.5
agent 2	0.34	0.16	0.5
agent 3	0.16	0.5	0.34

One possible FS1 allocation is:

$$\pi(1) = B, \quad \pi(2) = O_A, \quad \pi(3) = C$$

Agent 2 and 3 get utility 0.34, which exceeds their fair shares, which is $0.\bar{3}$ for all players. Agent 1 gets utility 0.2, but could get to 0.5 by receiving item A.
A possible Pareto improvement of the previous allocation is:

$$\pi(1) = o_A, \quad \pi(2) = C, \quad \pi(3) = B$$

In this case, agent 2 and agent 3 both get their fair shares. However, agent 1 gets utility 0.3, and even giving to him any of the small items in B and C he does not get his fair share.
Therefore, the new allocation, even though is a Pareto improvement of the previous one, does not satisfy FS1.

Hence, finding Pareto improvements that satisfies FS1 is a non trivial task. However, the same does not hold for FS property, which instead is preserved under Pareto improvement:

Proposition 3. *An allocation π' which is a Pareto improvement of a FS allocation π, satisfies FS*

Proof. Consider a FS allocation π and let π' be an allocation which Pareto dominates π. By FS we have:

$$u_i(\pi(i)) \geq \frac{1}{n} u_i(O) \quad \forall i \in N$$

Since π' is a Pareto improvement of π, we have:

$$u_i(\pi'(i)) \geq u_i(\pi(i)) \quad \forall i \in N$$

Therefore:

$$u_i(\pi'(i)) \geq \frac{1}{n} u_i(O) \quad \forall i \in N$$

Hence π' satisfies FS as well.

□

For now we will then focus on finding Pareto improvements without thinking about FS1 and relying on the preservation of FS. Instead, we can ask for a different requirement for our procedure for finding Pareto improvements, but, before specifying it, we need to introduce some basic concepts of graph theory. In particular, we focus on the definitions of *acyclic graph* and *complete bipartite graph*:

Definition 11. A *cycle* is a non-empty path in which only the first and last vertices are equal. A graph is said to be *acyclic* if it contains no graph cycles.

Definition 12. A *complete bipartite graph* is a graph whose vertices can be partitioned into two subsets V_1 and V_2 such that no edge has both endpoints in the same subset (*bipartiteness*), and every possible edge that could connect vertices in different subsets is part of the graph (*completeness).*

Now, we are able to introduce the concept of *consumption graph* related to a particular allocation:

Definition 13. Given a fractional or integral allocation x, the corresponding *consumption graph* G_x is a bipartite graph with vertices $N \cup O$ and edge set $E = \{\{i, o\} : x_{i,o} > 0\}$.

Essentially, it is a graph with agents and items as vertices: if an agent owns a share of an item, there is an edge between the agent's vertex and the item's one. Moreover, if an agent i shares an item with an agent j, we say that j is agent i's *neighbor.*

Finally, we can introduce the algorithm anticipated above, that given an FS fractional allocation y, returns an allocation x which Pareto improves y and it is FPO, preserving FS.

Algorithm 3 Algorithm for finding Pareto improvements: $\mathcal{A}_{imp}$

```
1: Define $G_x$ as the complete bipartite graph
2: Define an empty set of edges $T = \emptyset$
3: while there is a cycle C in the consumption graph $G_x$ do
4:     Solve the linear programming problem with value $\text{opt}_T$:
5:
6:     $\max \sum_{i \in N} (\sum_{o \in O} u_i(o) x_{i,o})$ such that:
```

$$\begin{cases} \sum_{o \in O} u_i(o) x_{i,o} \geq \sum_{o \in O} u_i(o) y_{i,o} & \forall i \in N \\ \sum_{i \in N} x_{i,o} = 1 \quad \forall o \in O \\ x_{i,o} = 0 \quad \forall (i,o) \in T \\ x_{i,o} \geq 0 \quad \forall i \in N \quad \text{and} \quad o \in O. \end{cases} .$$

```
7:     if there exists some $(i,o) \in C$ such that $\text{opt}_T = \text{opt}_{T \cup \{(i,o)\}}$ then
8:         $T = T \cup \{(i,o)\}$
9: return $x = x*$
```

The rationale behind the algorithm is essentially explained by the linear programming problem: we are looking for the allocation that maximises the sum of utilities, subjected to FS. Indeed, we have seen that Pareto improvements preserve FS: since the initial allocation is FS, we just need to find Pareto improvements.
In addition to the FPO and FS properties we obtain the additional result that the consumption graph G_{x^*} of the resulting allocation is acyclic, thanks to the while condition in step 3. In particular, the existence of an allocation x' such that all agents get the same utilities and the graph $G_{x'}$ is a subgraph of G_x but does not contain some edge from C, is granted by the following lemma:

Lemma 4. *For any allocation x, there is a FPO allocation x' such that:*

- *x' either Pareto dominates x or gives every agent the same utility as x*
- *the undirected consumption graph $G_{x'}$ is acyclic.*
- *x' has at most $n-1$ shared objects*

The last property is equivalent to say that $G_{x'}$ is a subgraph of G_x, since a shared object in the allocation x implies a link in the corresponding consumption graph.
The acyclicity of the resulting allocation's consumption graph is the additional requirement we needed for our procedure for finding Pareto improvement, since, as we will see in the next subsection, this property will be essential to find an FS1 and FPO allocation.

6.3 Algorithm to find a weighted FS1 and FPO allocation

In this subsection we are going to describe an algorithm that allows to find FS1 and PO allocation, in the general setting of weighted agents. The basic idea is to start with finding a fractional allocation that satisfies FS and to find a fractional and Pareto optimal allocation that Pareto improves the initial allocation. Then, the key step is to round the resultant fractional allocation to an integral one. In doing so, we deal with the consumption graph, and in particular, we require its acyclicity, which is granted by the use of the procedure described in the previous subsection. Hence, the Pareto improvement of the initial allocation is found by the use of $\mathcal{A}_{imp}$. Before explaining the rounding procedure, we formally describe the steps of the algorithm:

Algorithm 4 Algorithm to find a weighted FS1 and FPO allocation

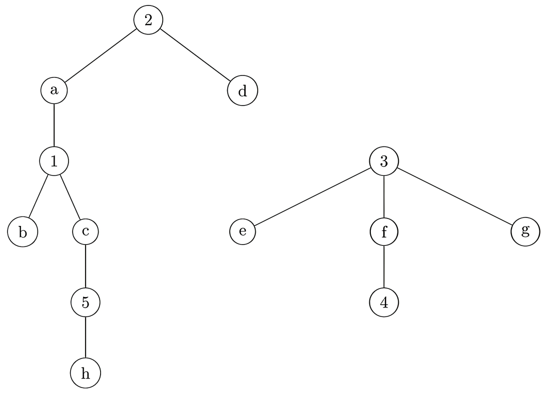
```
 1: Find a fractional allocation $x_{FS}$ that guarantees a share $b_i$ of each item to each
    agent $i$
 2: Using algorithm $\mathcal{A}_{impr}$, find an FPO fractional allocation $x$, with acyclic consump-
    tion graph $G_x$, that Pareto dominates $x_{FS}$
 3: if some agent $j$ shares an item $o$ for which $u_j(o) = 0$ then
 4:     $x_{j,o} = 0$ and $x_{i,o} = 1$ for some neighbour $i$ of $j$ (with the lowest index $i$)
 5: Consider $Q = \emptyset$, an empty FIFO queue of agents
 6: while there is an agent $i$ sharing at least one item $o$ with others do
 7:     Add agent $i$ (with the lowest index $i$) to $Q$
 8:     while $Q \neq \emptyset$ do
 9:         Take the first agent $j$ out of $Q$
10:         Add all the neighbours of $j$ to the end of $Q$ (lowest-index agents enter first)
11:         for each $o$ shared by $j$ do
12:             if $u_j(o) > 0$ then
13:                 give $o$ fully to $j$, i.e. $x_{j,o} = 1$ and $x_{k,o} = 0 \quad \forall k \neq j$
14:             else if $u_j(o) < 0$ then
15:                 give $o$ to a neighbour (with the lowest index $i$) with whom $o$ is shared
16: return the updated allocation $x^* = x$
```

The choices to select the agents with the lowest index (written inside the brackets) are just a tie-braking conventions.
As anticipated, the algorithm rounds the fractional allocation into an integral one. This is done starting from step 4. Essentially, the procedure is based on the acyclicity of the consumption graph: this ensures that the graph is a collection of trees. The rounding rule consists in picking an agent i, fully giving to him any good a he is sharing, and removing any chore b by his bundle, giving it to another owner of a share of b. Then, for any agent j who was sharing an item with i, the rule does the same, rounding all the fractions to his advantage, and so on. Looking at the consumption graph, this is equivalent to break the initial trees (starting at i) in other "subtrees" (starting from j), until no shared items are left; the termination of the algorithm is then ensured by acyclicity.
The following example should clarify the rounding procedure:

Example 5. Consider the following fractional allocation x, where we consider just if each item is a good (+) or a chore (−). The consumption of a particular item by an agent is indicated by a circle:

	a	b	c	d	e	f	g	h
agent 1	$\oplus$	$\oplus$	$\ominus$	+	-	+	+	+
agent 2	$\oplus$	-	-	$\oplus$	-	+	+	+
agent 3	+	+	-	-	$\ominus$	$\oplus$	$\oplus$	-
agent 4	-	+	-	-	-	$\oplus$	+	-
agent 5	-	-	$\ominus$	+	-	-	+	$\oplus$

The corresponding consumption graph is given by:



The rounding procedure, in this case, picks agent 2 first and gives him a (since it is a good) and d. Then, it continues with agent 1, giving to him item b and removing c from his bundle, since it is a chore. Item c is finally allocated to agent 5, who gets h as well.
Once the rounding is done in the first component, the rule proceeds with agent 3, in the second one: the only shared item is f, which is given to agent 3 because it is a good. Therefore, the rounding procedure obtains the following integral allocation:

	a	b	c	d	e	f	g	h
agent 1	+	$\oplus$	-	+	-	+	+	+
agent 2	$\oplus$	-	-	$\oplus$	-	+	+	+
agent 3	+	+	-	-	$\ominus$	$\oplus$	$\oplus$	-
agent 4	-	+	-	-	-	+	+	-
agent 5	-	-	$\ominus$	+	-	-	+	$\oplus$

Now that is clear how the algorithm works, we need to prove that it achieves the expected result. This is guaranteed by two lemmas.

Lemma 5. *Algorithm 4 returns an integral allocation that is weighted FS1*

Proof. First, we know that the allocation x returned in step 2 is a Pareto improvement of a FS allocation; hence, it satisfies FS as well. Clearly, moving null items does not affect such property, so steps 3 and 4 preserve FS as well.
Now, consider the two nested while cycles: the outer while cycle allows to consider the

different components of the consumption graph, the inner one explores inside a particular component. In particular, the reallocation involves items that are shared by more agents: no item fully consumed by one particular agent is reallocated. Hence, to prove that the final allocation is FS1, we just need to show that any agent j at the end loses at most one shared good and gets at most one shared chore (i.e., items o such that $x_{j,o} \in (0,1)$) and that the two cases are mutually exclusive.
We denote agent j picked by the internal while loop as an *active* agent: after the for cycle in step 11 no reallocation of items of agent j takes place, while during the cycle his utility can only increase. Hence, any possible decrease in utility of j happens before j becomes active. In particular, this means that before the for loop, an item was removed by j's bundle because it was a good shared with another agent i who considers it a good as well, or was a chore for both agents i and j and was allocated to j because he was the neighbour of i with the lowest index. In both cases, we denote agent i as the *predecessor* of i.
Since the consumption graph is acyclic, j can have at most one predecessor. This can be proven by contradiction. Suppose there are two predecessors of agent j, i and i'; then there is a path in the original consumption graph connecting i and i' (inside the while at step 8, only neighbours of previous active agents enter in Q), but since j's activation is after the activation of i and i', that path cannot pass through j; hence the original consumption graph contains a cycle (a path from i to i' not passing through j, and a path through j), a contradiction.
Finally, the outer while cycle runs until there are no shared items and each iteration reallocates at least one shared item: the algorithm terminates in a finite number of iterations and the result is an integral allocation, because of the while condition. Therefore, since each agent has at most one predecessor, x^* satisfies FS1. □

The second lemma is about the efficiency of the final allocation. Before stating it and proving it, we need to introduce a quantity named *weighted welfare*, which is $\sum_{i \in N} \lambda_i u_i(x_i)$ for some weighting $\lambda = (\lambda_i)_{i \in N}$, with $\lambda_i > 0 \quad \forall i \in N$. Intuitively, it is just a measure of the total satisfaction of all the players about a particular allocation x. Note that weighted welfare is not unique.
Thanks to the following result, which we will not prove, the concept of weighted welfare is strongly related to the efficiency of allocations:

Theorem 6. *An allocation is FPO if and only if it maximizes a weighted welfare*

Now, we are ready to state and prove the second lemma:

Lemma 7. *Algorithm 4 returns an integral allocation that is FPO*

Proof. The Pareto improvement x of the initial allocation is FPO; hence, by the previous theorem, it maximises a weighted welfare $\sum_{i \in N} \lambda_i u_i(x_i)$, with $\lambda_i > 0 \quad \forall i \in N$. After we apply the rounding procedure, the resulting consumption graph G_{x^*} is a subgraph of G_x and each item o is consumed by agents with the highest weighted utility $\lambda_i u_i(o)$. Hence x^* maximises the same weighted welfare of x. We conclude that x^* is FPO. □

Finally, putting the two lemmas together, we can state the main result of this section:

Theorem 8. *For mixed utilities, there always exists an integral allocation that satisfies weighted FS1 and FPO.*

7 Finding a connected FS1 allocation

Since FS1 is a weaker property than EF1, it is suitable to be required along with other constraints. In the previous section we added efficiency as an additional requirement for a suitable allocation, in this one we will look for a different property: *connectedness.* Formally, an allocation π is said to be *connected* if for each agent $i \in N$, $\pi(i)$ is connected in the sequence (*path*) of items $(o_1, o_2, ..., o_m)$.
In many applications, a disconnected division is not very useful: when we are dealing with land division, it is obviously a better solution for contingent lands to be owned by the same owner; the same is true for the task of allocating time slots to workers. Another example which need a connected solution could be the case where the items to be allocated are rooms, and the agents are a group of students who want adjacent rooms because they know each other.
Within this framework, we need to introduce another property, which is an extension of FS1 and takes into account the requirement of connectedness: FS1$_{outer}$. Formally:

Definition 14. A connected allocation π is *FS1$_{outer}$* if $\forall i \in N$:

- $u_i(\pi(i)) \geq \frac{1}{n}u_i(O)$ (the allocation is FS), or
- $u_i(\pi(i)) + u_i(o) \geq \frac{1}{n}u_i(O)$ for some $o \in O \setminus \pi(i)$ s. t. $\pi(i) \cup \{o\}$ is connected, or
- $u_i(\pi(i)) - u_i(o) \geq \frac{1}{n}u_i(O)$ for some $o \in \pi(i)$ s. t. $\pi(i) \setminus \{o\}$ is connected.

7.1 Generalised moving knife algorithm for divisible manna

Before moving to the indivisible case, we can look for a connected fair division in the cake-cutting setting of the first section, with an additional hypothesis on the utilities (uniformity on integral intervals), that in some sense shows some analogy to an indivisible one, as we will see later in this section.
First, we consider as a mixed cake the interval $[0, m]$, with $m \in \mathbb{N}$. The utility are taken to be uniform with value $u_i(o_j)$ in the interval $[j-1, j]$, with $j \in \{1, 2, ..., m\}$. Therefore, considering additive utilities, the fair share of agent i is $\frac{1}{n}\hat{u_i}([0, m])$ (with $\hat{u_i}$ we denote the utility defined on every interval, with $u_i(o_j)$ we denote the values taken by the utility function in the integral intervals) . The task is to find *contiguous allocations*, i.e., assignments of disjoint sub-intervals of the mixed cake to each agent such that the union of the intervals covers all the cake; moreover, we require that the allocations satisfies FS.
The solution to the problem is feasible and can be found generalising an algorithm which works with positive utilities: the *Dubins-Spanier moving knife procedure.* The rule, in the illustrative example of cake-cutting, is the following, and guarantees a FS contiguous allocation in the case of good manna:

1. A knife is slowly moved at constant speed parallel to itself over the top of the cake, covering an increasing sub-interval of the cake.
2. As soon as a slice that is equal to the fair share of any of the players is reached by the knife, that player stops the knife and receives that slice.

3. The process is repeated with the other $n-1$ agents and with what remains of the cake

4. The last player receives the remaining slice of cake.

Now, we can extend the rule to case of goods and chores, in what it is called the *generalised moving knife* algorithm. A moving knife procedure is applied to all players with strictly positive fair share, moving the knife from left to right in the interval $[0, m]$. If no agent with positive fair share exists, the knife is moved from right to left, until one agent stops the knife, as soon as the left part of the cake gives him his fair share; he receives that slice and the procedure is repeated on the remaining agents. Formally, given an allocation π, an agent set $N' \subseteq N$, and a sub-interval $[l, r]$ (we denote with $\pi|_{N'} : N' \longrightarrow 2^O$ the *restriction* of π to N', i.e. $\pi|_{N'}(i) = \pi(i) \quad \forall i \in N'$):

Algorithm 5 Generalised Moving Knife $\mathcal{A}$

```
1:  $\pi(i) = \emptyset \quad \forall i \in N'$
2:  $N^+ = \{i \in N' : \hat{u}_i([l,r]) > 0\}$
3:  if $N^+ \neq \emptyset$ then
4:      if $|N^+| = 1$ then
5:          Allocate $[l,r]$ to the unique agent in $N^+$
6:      else
7:          Let $x_i$ be the minimum point where $\hat{u}_i([l,x_i]) = \frac{1}{|N'|}\hat{u}_i([l,r])$ for agent $i \in N^+$
8:          Let agent j be the one with $x_j = \min_{i \in N^+} x_i$
9:          $\pi(j) = [l, x_j]$ and $\pi|_{N' \setminus \{j\}} = \mathcal{A}([x_j, r], N' \setminus \{j\}, (\hat{u}_i)_{i \in N' \setminus \{j\}})$
10:         return $\pi$
11: else
12:     Let $x_i$ be the maximum point where $\hat{u}_i([l,x_i]) = \frac{1}{|N'|}\hat{u}_i([l,r])$ for agent $i \in N'$
13:     Let agent j be the one with $x_j = \max_{i \in N'} x_i$
14:     $\pi(j) = [l, x_j]$ and $\pi|_{N' \setminus \{j\}} = \mathcal{A}([x_j, r], N' \setminus \{j\}, (\hat{u}_i)_{i \in N' \setminus \{j\}})$
15:     return $\pi$
```

Now, we can state the following theorem:

Theorem 9. *For additive utilities, a contiguous FS allocation of a mixed cake exists.*

Proof. We need to prove that the result of the algorithm satisfies the properties of theorem 3. In particular, given a mixed cake $[l, r]$, we will prove by induction on the number of agents $|N'|$ the following:

- if $N^+ \neq \emptyset$, all agents in N^+ receive at least their fair share $\frac{1}{|N'|}\hat{u}_i([l, r])$ and agents not in N^+ receive an empty piece;
- if $N^+ = \emptyset$ each agent receives at least their fair share $\frac{1}{|N'|}\hat{u}_i([l, r])$

First, when $|N'| = 1$ the claim is obviously true.
Now, suppose the algorithm returns an allocation with the desired properties for $|N'| = k - 1$.
If $N^+ \neq \emptyset$, the agents not in N^+ get nothing: hence, the only thing to prove is that the

agents in N^+ get their fair share. We assume that $|N^+| \neq 1$, since it is a trivial case. By construction, agent j receives his fair share, while all the other agents have at most utility equal to their fair share in the piece obtained by j; otherwise at least one of them would have stopped the knife before agent j, in contradiction to the minimality of x_j. Hence, each remaining agent i in N^+ has at least $\frac{(k-1)}{k}\hat{u}_i([l,r])$ utility left in sub-interval $[x_j, r]$. Moreover, we know by induction hypothesis that the algorithm works for $k-1$ agents, hence for each agent $i \neq j$:

$$\hat{u}_i(\pi(i)) \geq \frac{1}{k-1}\hat{u}_i([x_j,r]) \geq \frac{1}{k-1}\frac{(k-1)}{k}\hat{u}_i([l,r]) = \frac{1}{k}\hat{u}_i([l,r])$$

Therefore, each agent in N^+ receives his fair share, and the other agents get nothing. Now consider the case where no agent has a positive fair share. For the reason above, by maximality of x_j each agent has at most utility equal to his fair share in $[l, x_j]$: all agents different from j have at least $\frac{(k-1)}{k}\hat{u}_i([l,r])$ for the rest of the cake in $[x_j, r]$. Therefore, by induction hypothesis, each agent gets an interval of utility at least equal to his fair share. □

7.2 FS1 connected allocations

In the previous subsection we showed how the a contiguous FS allocation can be found in the case of divisible manna; however, the result can be used to prove that a contiguous FS1 allocation exists in the indivisible case, as stated in the following theorem:

Theorem 10. *For additive utilities, a connected FS1$_{outer}$ allocation of a path always exists*

Proof. To prove the existence, we adapt the rule of the previous subsection. In particular, given a path $(o_1, o_2, ..., o_m)$, we consider a mixed cake $[0, m]$ and uniform utilities $\hat{u}_i(o_j)$ in the interval $[j-1, j]$ for each $j \in (1, 2, ..., m)$; theorem 9 guarantees the existence of a contiguous and FS allocation. Without loss of generality, we assume that in such allocation $\hat{\pi}$ each agent $i = 1, 2..., n$ receives the sub-interval $[x_{i-1}, x_i]$ of the mixed cake $(x_0 = 0, x_n = m)$, and that no agent gets an empty bundle ($x_{i-1} < x_i$ for all $i = 1, 2..., n$).
Now, we can "translate" the divisible problem in the indivisible one, allocating each item o_j with the following rule: item o_j is assigned to agent i if $x_{i-1} \leq j-1 \leq j \leq x_i$ for some $i \in N$; otherwise it is allocated according to the preference of the left-most and right-most agents. The second case must be clarified, so we suppose that $j-1 \leq x_l \leq x_{l+1} \leq ... \leq x_r \leq j$, with $x_l = \min\{x_i : x_i \geq j-1\}$ (left-most agent) and $x_r = \max\{x_i : x_i \leq j\}$ (right-most agent). Now, we assign the item with the following rule:

- if $\min\{u_l(o_j), u_r(o_j)\} < 0 < max\{u_l(o_j), u_r(o_j)\}$ (one among agent l and agent r considers the item a chore, the other a good), then item o_j is allocated to the one with positive utility.
- if $\min\{u_l(o_j), u_r(o_j)\} \geq 0$, both agents have non-negative utility from o_j and this is assigned to the left-agent l.

- if $\max\{u_l(o_j), u_r(o_j)\} < 0$, both agents have negative utility from o_j and this is assigned to the right-agent r.

Note that if in the initial allocation one item is divided by three agents, the agent in the middle gets nothing in the final allocation. Now we show that the resulting allocation π is FS1$_{outer}$.
Consider an agent i, and first suppose that $x_{i-1}, x_i \notin \{1, 2, ..., m\}$ and $|x_i - x_{i-1}| > 1$. Let o_r and o_l be the boundary items, i.e, $x_{i-1} \in (r-1, r)$ and $x_i \in (l-1, l)$, while $\{o_{l+1}, o_{l+2}, ..., o_{r-1}\} \subseteq \pi(i)$, since they are completely contained in agent i's bundle. Now, we consider the 4 different cases:

- If $\min\{u_i(o_l), u_i(o_r)\} \geq 0$, i.e., both o_l and o_r are goods or null items for agent i, he receives at least o_r. Hence, if $o_l \in \pi(i)$, agent i gets his fair share; otherwise, he could get it by receiving the contiguous item o_l.

- If $\max\{u_i(o_l), u_i(o_r)\} < 0$, i.e., both o_l and o_r are chores for agent i, he does not receive o_r. Hence, if $o_l \notin \pi(i)$, agent i gets his fair share; otherwise, he could get it by removing the contiguous item o_l from his bundle.

- If $u_i(o_l) \geq 0$ and $u_i(o_r)$ <0, agent i does not receive o_r. Hence, if $o_l \in \pi(i)$, agent i gets his fair share; otherwise, he could get it by receiving the contiguous item o_l.

- If $u_i(o_l) < 0$ and $u_i(o_r)$ ≥0, agent i receives at least o_r. Hence, if $o_l \notin \pi(i)$, agent i gets his fair share; otherwise, he could get it by removing the item o_l from his bundle.

Therefore, in all 4 cases, agent i receives his fair share or could get it by adding or removing one of the boundary items: the bundle is then FS1$_{outer}$.
Now consider the other limit cases: if x_i or x_{i-1} is an integer, it is easy to see that π satisfies FS$_{outer}$. If $[x_{i-1}, x_i] \subseteq [j-1, j]$ for some j, agent i gets his fair share by receiving o_j or the empty bundle (if his fair share is non-positive).
The conclusion is that π is a connected FS1$_{outer}$ allocation.

□

8 Conclusions

In this work we studied the concept of fairness in fair division problems, introducing the properties of fair share and envy-freeness. Starting from two historical examples, we described how they were introduced in the setting of divisible manna.
We observed then that these properties must be modified to obtain feasible solutions in the indivisible case, and how the properties of envy-freeness up to one item and fair share up to one item are more flexible when dealing with such setting. Finally, we introduced the concept of Pareto optimality to offer a formal description of the efficiency of an allocation.
We then began looking for allocations satisfying different requirements, starting with the study on how the double round-robin rule accomplishes to find an EF1 allocation in the general case of mixed manna and fixed number of agents.
We discussed how the only case we could find a procedure satisfying efficiency along with EF1 is the one with 2 agents. Hence, we described the generalised adjusted winner, an algorithm to find such an allocation, which extended the adjusted winner algorithm to the case of mixed manna.
Noting that FS1 is a weaker requirement than EF1, we looked for a procedure to find efficient allocations guaranteeing FS1 for any number of agents. This was found to exist, basing its rationale on the rounding, through the modification of the related consumption graph, of a fractional allocation satisfying FS and PO. This in turn was obtained with a particular algorithm for finding Pareto improvements of a FS allocation, with the additional requirement of acyclicity for the resultant consumption graph.
Finally, we studied the problem of finding connected allocations, with the introduction of the FS1_{outer} property. An allocation with such requirements was found by translating the solution (respecting connectedness) of a divisible problem into an indivisible one.

9 References

[1] Haris Aziz, Hervé Moulin, and Fedor Sandomirskiy. “A polynomial-time algorithm for computing a Pareto optimal and almost proportional allocation”. en. In: *Operations Research Letters* 48.5 (Sept. 2020), pp. 573–578. ISSN: 01676377. DOI: 10.1016/j.orl.2020.07.005. (Visited on 06/29/2023).

[2] Haris Aziz et al. “Fair allocation of indivisible goods and chores”. en. In: *Autonomous Agents and Multi-Agent Systems* 36.1 (Apr. 2022), p. 3. ISSN: 1387-2532, 1573-7454. DOI: 10.1007/s10458-021-09532-8. (Visited on 06/29/2023).

[3] Vittorio Bilò et al. “Almost envy-free allocations with connected bundles”. en. In: *Games and Economic Behavior* 131 (Jan. 2022), pp. 197–221. ISSN: 08998256. DOI: 10.1016/j.geb.2021.11.006. (Visited on 06/29/2023).

[4] Steven J. Brams and Alan D. Taylor. “An Envy-Free Cake Division Protocol”. In: *The American Mathematical Monthly* 102.1 (1995), pp. 9–18. ISSN: 00029890, 19300972. (Visited on 07/01/2023).

[5] L. E. Dubins and E. H. Spanier. “How to Cut A Cake Fairly”. In: *The American Mathematical Monthly* 68.1 (Jan. 1961), p. 1. ISSN: 00029890. DOI: 10.2307/2311357. (Visited on 06/29/2023).

[6] Hervé Moulin. “Fair Division in the Internet Age”. en. In: *Annual Review of Economics* 11.1 (Aug. 2019), pp. 407–441. ISSN: 1941-1383, 1941-1391. DOI: 10.1146/annurev-economics-080218-025559. (Visited on 06/29/2023).

[7] Fedor Sandomirskiy and Erel Segal-Halevi. “Efficient Fair Division with Minimal Sharing”. en. In: *Operations Research* 70.3 (May 2022), pp. 1762–1782. ISSN: 0030-364X, 1526-5463. DOI: 10.1287/opre.2022.2279. (Visited on 06/29/2023).

[8] Hal R. Varian. “Two problems in the theory of fairness”. In: *Journal of Public Economics* 5.3 (1976), pp. 249–260. ISSN: 0047-2727. DOI: https://doi.org/10.1016/0047-2727(76)90018-9.