

Fairly Allocating (Contiguous) Dynamic Indivisible Items with Few Adjustments

Mingwei Yang
Peking University
yangmingwei@pku.edu.cn

Abstract

We study the problem of dynamically allocating indivisible items to a group of agents in a fair manner. We assume that the items are goods and the valuation functions are additive without specification. Due to the negative results to achieve fairness, we allow adjustments to make fairness attainable with the objective to minimize the number of adjustments. We obtain positive results to achieve EF1 for (1) two agents with mixed manna, (2) restricted additive or general identical valuations, and (3) the default setting. We further impose the contiguity constraint on the items and require that each agent obtains a consecutive block of items. We obtain both positive and negative results to achieve either EF1 or proportionality with an additive approximate factor. In particular, we establish matching lower and upper bounds to achieve approximate proportionality for identical valuations. Our results exhibit the large discrepancy between the identical model and nonidentical model in both contiguous and noncontiguous settings. All our positive results are computationally efficient.

1 Introduction

Fair division is one of the most fundamental and well-studied topics in Computational Social Choice with much significance and several applications in many real-life scenarios [17, 27]. Generally, there are some resources and our objective is to divide them among a group of competing agents in a fair manner. Arguably, the most compelling fairness notion is *envy-freeness*, which is defined as each agent weakly preferring his own bundle to any other agent’s bundle.

However, in the indivisible regime, the existence of envy-free solutions is not guaranteed. For instance, if there are two agents but only one item, the agent who receives the item is certainly envied by the other one. One of the natural relaxations of envy-freeness is *envy-freeness up to one item (EF1)* proposed by [13]. In an EF1 allocation, any agent i may envy another agent j , but the envy could be eliminated by removing one item from agent j ’s bundle. EF1 allocations not only exist in any instance but also can be computed in polynomial time [21].

Imposing some constraints to the model will make the fairness objective less tractable. A series of works focus on the setting where the items lie on a line and each agent obtains a consecutive block of items. For monotone valuation functions, [9] presents polynomial-time algorithms to compute contiguous EF1 allocations for any number of agents with identical valuations or at most three agents, and shows the existence of contiguous EF1 allocations for four agents. Not until recently, the existence of contiguous EF1 allocations for any number of agents is proved by [20].

Another natural generalization assumes that the items arrive online. Furthermore, the types of future items are unknown and the decisions have to be made instantly. Recently, [18] investigates this model with the requirement that the allocations returned after the arrival of each item are EF1.

However, due to the negative results against the adversary [8], adjustments are necessary to achieve EF1 deterministically. Notably, the $O(T^2)$ of adjustments is attained trivially by redistributing all items in each round, where T is the number of items. [18] shows that the $\Omega(T)$ of adjustments is inevitable for more than two agents, even if the information of the items is known upfront. On the positive side, they give two algorithms with respectively $O(T^{1.5})$ and $O(nmT)$ adjustments, where m denotes the maximum number of distinct valuations for a single item among all agents.

In this work, we adopt the online model proposed by [18]. Furthermore, we also impose the contiguity constraint on the items. Specifically, the items are arranged on a line by the order they arrive and each agent obtains a contiguous block of items. To motivate this, consider a library that has several bookshelves with books of the same types being placed together. Moreover, the number of books of a certain type should not differ by a large amount. With more and more bookshelves being deployed, the library needs to reallocate a consecutive block of bookshelves to a certain type of books and redistribute the books according to the new allocation. In this case, the objective is to minimize the cost of moving the books or, equivalently, the number of adjustments. We study the number of adjustments necessary to achieve some fairness guarantee in both contiguous and noncontiguous models.

1.1 Our Contributions

We say that an algorithm is fair concerning some fairness notion if the allocations returned by the algorithm satisfy such notion. Without further specification, we assume that items are goods and the valuation functions are additive.

We first demonstrate our positive results to achieve EF1 in the noncontiguous model. For two agents, we extend the $O(T)$ upper bound given by [18] for goods to the mixed manna case [22], in which items may have both positive and negative valuations. Then we show that if the valuation functions are limited to be restricted additive [1] or general identical, EF1 comes for free in the sense that no adjustments are required. In addition, we give an EF1 algorithm that requires $O(mT)$ adjustments. Note that if m is a constant, it matches the $\Omega(T)$ lower bound and thus is optimal.

With the contiguity constraint, EF1 is too stringent and we start with a weaker fairness notion. [28] shows that the contiguous $(\frac{n-1}{n} \cdot v^{\max})$ -proportional allocation can be computed efficiently, where v^{\max} is the maximum valuation of a single item, and this additive approximate factor is tight in some sense. Denote $(\frac{n-1}{n} \cdot v^{\max})$ -proportionality as PROP_a. When valuations are identical, we give a PROP_a algorithm that requires $O(nT)$ adjustments. Then we establish the matching lower bound to show that our algorithm is optimal. When it comes to EF1, for two agents with general valuations, we show that EF1 is achievable with $O(T)$ adjustments, which is optimal. If the valuation of each item is further assumed to lie in $[L, R]$ for $0 < L \leq R$, we give an EF1 algorithm that requires $O(\frac{R}{L} \cdot n^2 T)$ adjustments. By contrast, in the nonidentical setting, we show that it is impossible to make any significant improvement. Specifically, we give instances to establish the lower bounds of $\Omega(T^2/n)$ to achieve PROP_a and $\Omega(T^2)$ for EF1. Even in the binary case where the valuation of each item is either 0 or 1, we prove that for two agents, the $\Omega(T^2)$ lower bound still pertains to achieving PROP_a. Our results in the contiguous model are summarized in Table 1.

Our results exhibit the large discrepancy between the identical case and nonidentical case in both contiguous and noncontiguous settings. In addition, all the algorithms given in this work are computationally efficient.

Table 1: Results for the contiguous model. Here we only demonstrate our results for additive valuations. Particularly, the $O(T)$ upper bound in the identical setting for two agents works for general valuations, and the $\Omega(T)$ lower bound in the nonidentical setting for two agents works for binary valuations.

| | Identical | | | Nonidentical | |
|---------|-------------------|--------------|--|-------------------|---------------|
| | PROP _a | EF1 | | PROP _a | EF1 |
| | | Lower | Upper | | |
| $n = 2$ | | $\Theta(T)$ | | $\Theta(T^2)$ | |
| $n > 2$ | $\Theta(nT)$ | $\Omega(nT)$ | $O\left(\frac{R}{L} \cdot n^2T\right)$ | $\Omega(T^2/n)$ | $\Theta(T^2)$ |

1.2 Related Work

Even though both divisible and indivisible models are extensively studied, we only focus on the indivisible setting, which is more relevant to our work.

Dynamic fair division. Our work belongs to the vast literature of *dynamic* or *online* fair division [4]. Under the assumption that valuations are normalized to $[0, 1]$ and the items are allocated irrevocably, [8] gives an algorithm that achieves maximum envy of $\tilde{O}(\sqrt{T/n})$ deterministically and shows that this bound is tight asymptotically. To bypass the negative results, motivated by the notion of *disruptions* [14, 15], [18] introduces adjustments to achieve EF1 deterministically.

Without the ability to adjust, another popular measure of compromise is assuming that agents' valuations are stochastic. When the valuation of each agent to each item is drawn i.i.d. from some distribution, the algorithm of allocating each item to the agent with the maximum valuation for it is envy-free with high probability and ex-post Pareto optimal [23, 24]. [6] provides the same guarantee for asymmetric agents, i.e., the valuation of each agent is independently drawn from an agent-specific distribution. [29] further assumes that the valuations of different agents for the same item are correlated and shows that Pareto efficiency and fairness are also compatible. More recently, [7] studies the partial-information setting where only the ordinal information is revealed. Another series of works [2, 3] resort to random allocations to achieve ex-ante fairness together with some efficiency and incentive guarantees.

Fair division of contiguous blocks. The online fair division model with the contiguity requirement concerned in our work is a strict extension of the fair division of contiguous blocks problem. In the offline setting, [9, 20, 26] study the existence of contiguous EF1 allocations and [28] focuses on the approximate versions of proportionality, envy-freeness, and equitability. More recently, [25] designs an algorithm to compute a contiguous EQ1 allocation with the egalitarian welfare guarantee. Besides, the *price of fairness* of contiguous allocations for both goods and chores are respectively established by [28] and [19]. More generally, [10, 11] allow the connectivity relation among the items to form a graph that possesses some structures with the path a special case.

2 Preliminaries

There are n agents and let $N = [n]$ be the set of all agents. There are T items arriving one by one, one at a time. Let g_t be the t -th item and $M_t = \{g_1, \dots, g_t\}$ be the set of the first t items that arrive. In particular, let $M = M_T$. Each agent i has a valuation function $v_i : 2^M \rightarrow \mathbb{R}$, which

reflects his evaluation for each subset of items. To simplify the notations, for $S \subseteq M$ and $g \in M$, we use $v_i(g)$, $v_i(S + g)$ and $v_i(S - g)$ to respectively represent $v_i(\{g\})$, $v_i(S \cup \{g\})$ and $v_i(S \setminus \{g\})$. An allocation for a set of items M' is a tuple of sets (A_1, A_2, \dots, A_n) satisfying $A_i \cap A_j = \emptyset$ for all $i \neq j$ and $A_1 \cup A_2 \cup \dots \cup A_n = M'$, where A_i is agent i 's bundle.

After the arrival of each item, the algorithm should return an allocation for all the existing items. We use the number of adjustments to measure the performance of algorithms. Formally, let the allocations returned by an algorithm be A^1, \dots, A^T , where A^t is an allocation for M^t for all $t \in [T]$. The number of adjustments used in round $t > 1$ is defined as the number of items $g \in M_{t-1}$ such that g belongs to different agents in A^{t-1} and A^t . The number of adjustments required by an algorithm is defined as the total number of adjustments used across all T rounds.

Valuation functions. Now we describe different types of valuation functions that are considered in this work. A valuation function v is *additive* if for all $S_1 \subseteq M$ and $S_2 \subseteq M \setminus S_1$, we have $v(S_1 \cup S_2) = v(S_1) + v(S_2)$. We say that the valuations are *restricted additive* if they are additive, and furthermore, each item g has an inherent valuation v_g so that $v_i(g) \in \{0, v_g\}$ for any agent i . If a valuation function v is *binary*, then for any item g , we have $v(g) \in \{0, 1\}$. A valuation function v is called *general* if it is only required to be monotone, i.e., $v(S + g) \geq v(S)$ for all $S \subseteq M$ and $g \in M$. The valuations are *identical* if we have $v_1(\cdot) = \dots = v_n(\cdot)$. In this work, we assume the valuations to be additive without specification.

Categories of items. Items are classified into goods, chores, and mixed manna. Agents have nonnegative valuations for goods and nonpositive valuations for chores. An item being mixed manna means that it can be a good for some agents and, at the same time, a chore for some other agents. In this work, we assume all items to be goods without specification.

Now we define EF, EF1, and PROPa, which will be the fairness criteria in this work.

Definition 2.1 (EF). An allocation A is *envy-free (EF)* if for any $i, j \in N$, we have $v_i(A_i) \geq v_i(A_j)$.

Definition 2.2 (EF1). An allocation A is *envy-free up to one item (EF1)* if for any $i, j \in N$, there exists $g \in A_i \cup A_j$ such that $v_i(A_i - g) \geq v_i(A_j - g)$.

Notably, when the items are goods, the definition of EF1 can be simplified as taking away an item in agent j 's bundle. However, if there are chores or mixed manna, removing an item in agent i 's bundle may make agent i better off if the item is a chore for agent i .

Definition 2.3 (PROPa). Let $v^{\max} = \max_{i \in N, g \in M} v_i(g)$. An allocation A is $(\frac{n-1}{n} \cdot v^{\max})$ -proportional (PROPa) if we have

$$v_i(A_i) \geq \frac{1}{n} \cdot v_i(M) - \frac{n-1}{n} \cdot v^{\max}$$

for any $i \in N$.

3 Noncontiguous Setting

In this section, we present our positive results in the noncontiguous setting.

3.1 Two Agents and Mixed Manna

In this section, for two agents and mixed manna, we give an EF1 algorithm that requires $O(T)$ adjustments, which is demonstrated in Algorithm 1. It is the extension of the Envy Balancing Algorithm for two agents and goods given by [18], which assumes that we know the information of all items in advance and requires no adjustments.

The idea of Algorithm 1 is based on the envy-cycle elimination algorithm, which is proposed by [21] to efficiently compute an EF1 allocation. In Algorithm 1, we maintain two disjoint allocations G and C satisfying that G is EF and C is EF1. Since the combination of an EF allocation and an EF1 allocation is also EF1, the returned allocation, which is the combination of G and C , is also EF1. To this end, we only assign the new item to an agent in C while maintaining the EF1 property of C (Line 3–Line 7). Furthermore, when both agents envy each other in C , we swap the bundles in C to make C an EF allocation (Line 8–Line 10). Once C becomes EF, we merge C into G (Line 11–Line 14).

Algorithm 1: Envy Balancing Algorithm for Mixed Manna

```

Input:  $v_1, v_2$ 
1  $G \leftarrow (\emptyset, \emptyset), C \leftarrow (\emptyset, \emptyset)$ 
2 for  $t = 1, \dots, T$  do
3   if  $(a_1 \text{ is unenvied in } C \wedge v_1(g_t) > 0) \vee (v_1(g_t) > 0 \wedge v_2(g_t) \leq 0) \vee (a_1 \text{ does not envy}$ 
    $a_2 \text{ in } C \wedge v_1(g_t) \leq 0 \wedge v_2(g_t) \leq 0)$  then
4      $C \leftarrow (C_1 \cup \{g_t\}, C_2)$ 
5   else
6      $C \leftarrow (C_1, C_2 \cup \{g_t\})$ 
7   end
8   if  $a_1$  and  $a_2$  envy each other in  $C$  then
9      $C \leftarrow (C_2, C_1)$ 
10  end
11  if  $C$  is envy-free then
12     $G \leftarrow (G_1 \cup C_1, G_2 \cup C_2)$ 
13     $C \leftarrow (\emptyset, \emptyset)$ 
14  end
15   $A^t = (C_1 \cup G_1, C_2 \cup G_2)$ 
16 end
17 return  $[A^1, A^2, \dots, A^T]$ 

```

Theorem 3.1. *Algorithm 1 is an EF1 algorithm for two agents and mixed manna that requires $O(T)$ adjustments.*

Proof. First, we prove that every returned allocation is EF1. Since the combination of an EF allocation and an EF1 allocation is also an EF1 allocation, it suffices to prove that G is always EF and C is always EF1. Notice that we merge C into G only if C is EF, and the combination of two EF allocations is also EF. Thus G is always EF.

Now we prove that C is always EF1. We only need to show that, if C is EF1 at the beginning of the current iteration, then it is also EF1 at the end of the current iteration. Note that if C is EF1, then by definition, at least one of the agents is unenvied in C since otherwise the two bundles in C would have been swapped in the previous iteration, which makes C an EF allocation. Similarly,

there is at least one of the agents does not envy the other agent in C if C is EF1. After the new item comes, according to Line 3, if it is a good for an agent and a chore for the other agent, i.e., $v_1(g_t) > 0 \wedge v_2(g_t) \leq 0$ or $v_1(g_t) \leq 0 \wedge v_2(g_t) > 0$, then it will be assigned to the agent who views it as a good, which ensures that both $v_1(C_1) - v_1(C_2)$ and $v_2(C_2) - v_2(C_1)$ are nondecreasing. In this case, the EF1 property of C is maintained. If the new item is a good for both agents, i.e., $v_1(g_t) > 0 \wedge v_2(g_t) > 0$, then the agent who it is assigned to is unenvied by the other agent in C . Since no one will envy the agent who gets the new item after removing it, the newly generated C is EF1 by definition. Similarly, if the new item is a chore for both agents, i.e., $v_1(g_t) \leq 0 \wedge v_2(g_t) \leq 0$, then the agent who it is assigned to does not envy the other agent in C . Since the agent who gets the new item envies no one after removing it, the newly generated C is EF1 by definition.

Next, we prove the number of adjustments required by Algorithm 1. Note that adjustments happen only when both agents envy each other in C , and, in this case, their bundles in C are swapped, which contributes $|C_1 \cup C_2|$ adjustments. After the swap, C becomes EF and is merged into G . Since no adjustments are made on G , we conclude that each item is adjusted at most once. Therefore, the total number of adjustments is $O(T)$. \square

3.2 Restricted Additive or General Identical Valuations

For additive valuations, [18] proves that every EF1 algorithm requires at least $\Omega(T)$ adjustments. Surprisingly, we show that if we limit the valuations to restricted additive or general identical valuations, EF1 can be achieved with no cost: simple greedy algorithms is EF1 and require no adjustments for these two cases. Notably, restricted additive valuations can be viewed as a slightly more general version of additive identical valuations by adding the assumption that each agent may be indifferent to some items. Recalling that the lower bound for the nonidentical case is $\Omega(T)$, our results reveal the large discrepancy between the identical case and the nonidentical case in the noncontiguous setting.

First, we give the greedy algorithm for restricted additive valuations in Algorithm 2. The idea of Algorithm 2 is simply giving the new item to the agent k with minimum valuation among the agents with positive valuations for it. Without loss of generality, here we assume that for every item, there exists an agent that has a positive valuation for it. Since otherwise all agents are indifferent to the item and we just ignore it.

Algorithm 2: Greedy Algorithm for Restricted Additive Valuations

Input: v_i for each agent a_i

```

1  $A = \emptyset$ 
2 for  $t = 1, \dots, T$  do
3    $k \leftarrow \arg \min_{i: v_i(g_t) > 0} v_i(A_i)$ 
4    $A_k \leftarrow A_k \cup \{g_t\}$ 
5    $A^t \leftarrow A$ 
6 end
7 return  $[A^1, A^2, \dots, A^T]$ 
```

Theorem 3.2. *Algorithm 2 is an EF1 algorithm for restricted additive valuations that requires no adjustments.*

Proof. Since once an item is assigned to an agent, Algorithm 2 will keep allocating it to the same agent in the future, it is clear that no adjustments are made. Now we prove that every returned

allocation is EF1. First we prove the property that, for any $i, j \in N$, $v_i(A_j) \leq v_j(A_j)$ is always held. In other words, each bundle is valued the most by the agent who owns it. Since Algorithm 2 only assign an item to the agent who has positive valuation for it, for any agent j , we have $v_j(A_j) = \sum_{g \in A_j} v_g$, where v_g is the inherent valuation of item g . Since agent i may be indifferent to some items in A_j , we have $v_i(A_j) \leq \sum_{g \in A_j} v_g = v_j(A_j)$.

Next, we use induction for proof. Before the arrival of the first item, the allocation is clearly EF1. We will show that if the current allocation is EF1, then after the arrival of a new item, the newly generated allocation is also EF1. When a new item g comes, define $N_1 = \{i \in N \mid v_i(g) > 0\}$ as the set of agents who are interested in g and $N_2 = N \setminus N_1$ as the set of agents who are not. Since the agents in N_2 are indifferent to g , their existence is not likely to lead to the violation of EF1 no matter which agent g is assigned to. Therefore, we are only concerned about the agents in N_1 . Let $k \in N_1$ be the agent who gets g . By the choice of k , for any $i \in N_1$, we have $v_i(A_k) \leq v_k(A_k) \leq v_i(A_i)$ before the assignment of g , which means that each agent in N_1 does not envy agent k . Thus after allocating g to agent k , each agent in N_1 will not envy agent k up to one item. This concludes the proof. \square

Then we give an algorithm for general identical valuations in Algorithm 3. Similarly, the idea of Algorithm 3 is simply assigning the new item to the agent with minimum valuation.

Algorithm 3: Greedy Algorithm for General Identical Valuations

Input: v_i for each agent a_i

```

1  $A = \emptyset$ 
2 for  $t = 1, \dots, T$  do
3    $k \leftarrow \arg \min_i v_i(A_i)$ 
4    $A_k \leftarrow A_k \cup \{g_t\}$ 
5    $A^t \leftarrow A$ 
6 end
7 return  $[A^1, A^2, \dots, A^T]$ 
```

Theorem 3.3. *Algorithm 3 is an EF1 algorithm for general identical valuations that requires no adjustments.*

Proof. Since once an item is assigned to an agent, Algorithm 3 will keep allocating it to the same agent in the future, it is clear that no adjustments are made. Now we use induction to prove that every returned allocation is EF1. When there are no items, the allocation is clearly EF1. We will prove that if the current allocation is EF1, then after the arrival of a new item, the newly generated allocation is also EF1. When a new item g comes, suppose that agent k is the agent with minimum valuation and thus gets g . Since all agents do not envy agent k before the assignment by the choice of k , they will not envy agent k up to one item in the new allocation. This concludes the proof. \square

3.3 The Default Setting

Recall that for arbitrary number of agents with additive valuations, [18] shows a lower bound of $O(T)$ adjustments and gives two algorithms with respectively $O(T^{1.5})$ and $O(nmT)$ adjustments, where $m = \max_{i \in N} |\{v_i(g) \mid g \in M\}|$ is the maximum number of distinct valuations for items among all agents. Now we present our algorithm that requires $O(mT)$ adjustments in Algorithm 4. Note that when m is a constant, our algorithm is optimal.

The idea of the algorithm is based on the round-robin algorithm. Recall that in the round-robin algorithm, agents pick their favorite items alternately in each round. The resulting allocation is EF1 since in each round, every agent prefers the item that he picks in this round to any item picked by some agent in the next round. On the contrary, this is also a sufficient condition for EF1. Thus we maintain the above property for each allocation in our algorithm. To be more specific, we maintain a round-robin-like structure by dividing the items into multiple layers satisfying that each agent gets exactly one item in each layer except the last one, in which each agent gets at most one item. Moreover, it holds that in each layer, each agent prefers his item in this layer to any item in the next layer. In this case, the allocation induced by the round-robin-like structure is EF1. Similar arguments are used in [12] to show that the allocation induced by recursively finding a maximum-weight matching between agents and items is EF1.

Our algorithm is presented in Algorithm 4, which is the realization of the above idea. We use C^k to record the allocation of the k -th layer, where C_a^k denotes the item obtained by agent a in this layer. In particular, $C_a^i = 0$ if agent i does not obtain any item in round i , and let $v_i(0) = 0$ for all $i \in N$. When a new item comes, the algorithm updates each layer sequentially. Whenever there is an agent who prefers the new item to his item in the current layer, then we use the new item to replace his item in the current layer (Line 5–Line 8). After all the replacements in the current layer are finished, we then move to the next layer and repeat the above process. In particular, in the last layer, we assign the new item to an arbitrary agent who has not obtained an item in the last layer (Line 10). Finally, we return the allocation induced by all layers (Line 11).

Algorithm 4: Layer Updating Algorithm

Input: v_i for each agent a_i
1 $C_j^i \leftarrow 0$ for all $i = 1, \dots, \lceil \frac{T}{n} \rceil$ and $j = 1, \dots, n$
2 **for** $t = 1, \dots, T$ **do**
3 $k \leftarrow \lceil \frac{t}{n} \rceil$
4 **for** $i = 1, \dots, k - 1$ **do**
5 **while** $G := \{a \in N \mid v_a(g_t) > v_a(C_a^i)\} \neq \emptyset$ **do**
6 $\hat{a} \leftarrow \arg \min_{a \in G} C_a^i$
7 $\text{swap}(g_t, C_{\hat{a}}^i)$
8 **end**
9 **end**
10 $C_{t-n*(k-1)}^k \leftarrow g_t$
11 Let A^t be the combination of allocations C^1, C^2, \dots, C^k
12 **end**
13 **return** $[A^1, A^2, \dots, A^T]$

Theorem 3.4. *Algorithm 4 is an EF1 algorithm that requires $O(mT)$ adjustments.*

Proof. We use layer k to refer to allocation C^k . We first prove that throughout the algorithm, for each layer k , each agent prefers the item that he obtains in this layer to any item in layer $k + 1$, i.e., $v_i(C_i^k) \geq v_i(C_j^{k+1})$ for any $i, j \in N$. Then we will show that this condition is sufficient for EF1.

We first use induction to prove that

$$v_i(C_i^k) \geq v_i(C_j^{k+1}), \quad \forall i, j \in N, \forall k > 0. \quad (1)$$

When there are no items, (1) is trivially satisfied. We will show that if (1) holds for the current allocation, then after the arrival of a new item, (1) also holds for the newly generated allocation.

Note that during the update of a layer, each replacement will strictly increase the valuation of the chosen agent for his item in this layer. Thus after each replacement, the chosen agent still prefers his item in this layer to any item in the next layer. Moreover, the update for a layer being finished indicates that each agent prefers his item in this layer to the new item. As a result, we can safely move on and update the next layer with the new item.

Suppose that (C^1, C^2, \dots) satisfies (1) and let A be the allocation induced by (C^1, C^2, \dots) . Now we prove that A is EF1. Note that for any $i \in N$, we have $A_i = \{C_i^1, C_i^2, \dots\}$. Thus for any agents $i, j \in N$, we have

$$v_i(A_i) = \sum_{k \geq 1} v_i(C_i^k) \geq \sum_{k \geq 1} v_i(C_j^{k+1}) = v_i(A_j - C_j^1).$$

Therefore, we conclude that agent i does not envy agent j up to one item and A is EF1.

Finally, we prove that the algorithm uses $O(mT)$ adjustments in total. Notice that the number of adjustments is upper bounded by the number of replacements. Since a replacement will make the valuation of the chosen agent for his item in the current layer strictly larger and there are at most m distinct valuations of any agent for different items, the number of replacements throughout the algorithm for an agent in a certain layer is upper bounded by m . Therefore, there are at most $O(mT)$ adjustments. \square

Notice that when valuations are further assumed to be identical, the agents share identical preferences for items. If the update of a layer is going to happen, the item belonging to the chosen agent in this layer must be the least valued one among all items in this layer as well as the new item. Thus after the replacement, each agent will prefer his item in this layer to the new item and no more replacements will be made in this layer. Therefore, the number of adjustments in round t can be upper bounded by the number of layers, which is $O(t/n)$, and the number of adjustments required by the algorithm can be also upper bounded by $\sum_{t \in [T]} O(t/n) = O(T^2/n)$. In some special cases such that $n = \Omega(T^\alpha)$ for some $\alpha > 0$, it also allows us to establish an upper bound of $O(T^{2-\alpha})$ for identical valuations.

Corollary 3.5. *When the valuations are identical, Algorithm 4 is an EF1 algorithm that requires $\min(O(mT), O(T^2/n))$ adjustments.*

Remark 3.6. *All the existing positive results for goods can be extended to chores. For the greedy algorithms for restricted additive or general identical valuations, just let the agent with maximum valuation rather than minimum valuation obtain the new item in each round. Furthermore, all the positive results for any number of agents with additive valuations are based on the round-robin algorithm. By adding some worthless items to make the number of items a multiple of n , the allocation returned by the round-robin algorithm for chores also possesses the desirable properties [5], and thus we can prove the upper bounds analogously.*

Remark 3.7. *It is natural to ask about the incentive issue of the algorithms. That is, apart from the objective of fairness, we also hope that each agent is better off after the adjustments to make them happy with the new allocation in each round. In fact, it is not difficult to see that all our algorithms concerning goods in the noncontiguous setting satisfy the incentive property. Furthermore, if we re-run the round-robin algorithm with a deterministic tie-breaking rule upon the arrival of a new good, then we can prove that the valuation of each agent will not decrease. As a result, the incentive property also holds for the algorithms given by [18] which are based on the round-robin algorithm (see Appendix A for more discussion).*

By contrast, with the contiguity constraint, suppose that the order of the agents stays the same throughout the entire period, i.e., agent i obtains the i -th block, which is the default assumption for identical valuations in this work. Then the incentive property is not compatible with EF1, even for two agents with additive identical valuations. For instance, there are three items with valuations of 1, 3 and 2, respectively. In round 2, we have $A = (\{g_1\}, \{g_2\})$. However, in round 3, the only contiguous allocation that satisfies EF1 is $A = (\{g_1, g_2\}, \{g_3\})$, in which the valuation of agent 2 is less than the previous round.

4 Contiguous Setting with Identical Valuations

In this section, we assume that all items are arranged on a line by the order they arrive. Moreover, we impose the constraint on the allocation that it has to be contiguous, i.e., each agent gets a contiguous block of items. We assume valuations to be identical in this section, which can be represented by a valuation v shared by all agents, and agent i always gets the i -th block from left to right. Let $M_{l,r} = \{g_{l+1}, \dots, g_r\}$ be the set of items whose indexes are in $[l+1, r]$ with $M_{0,r} = M_r$. We will establish lower and upper bounds for both PROP_a and EF1.

4.1 Upper Bounds for PROP_a

In the offline model, for any line of items M and n agents with nonidentical valuation functions v_1, \dots, v_n , [28] designs an efficient algorithm to compute a contiguous allocation $A = (A_1, A_2, \dots, A_n)$ satisfying

$$v_i(A_i) \geq \frac{1}{n} \cdot v_i(M) - \frac{n-1}{n} \cdot v_i^{\max}(M) \quad (2)$$

for all $i \in N$, where $v_i^{\max}(M) = \max_{g \in M} v_i(g)$. In particular, the computed allocation is PROP_a. In fact, [28] also show that this is the best additive approximation factor even for identical valuations and without the contiguity requirement. A direct consequence of the algorithm is showing the existence of an allocation satisfying (2) for all $i \in N$.

Lemma 4.1 ([28]). *Suppose that there are n agents and the items in M are arranged on a line. Then there exists a contiguous allocation (A_1, A_2, \dots, A_n) satisfying (2) for all $i \in N$.*

We start by describing the offline algorithm. The algorithm maintains a current block, iterates all items sequentially, and adds the new item to the current block. In the beginning, all agents are active. Whenever there exists an agent i satisfying (2) for the current block, then the current block is obtained by agent i . After that, agent i is deactivated and the current block becomes empty. When all agents become inactive, the remaining items are assigned arbitrarily.

For identical valuations in the online setting, we give an algorithm based on the offline algorithm that achieves PROP_a with $O(nT)$ adjustments in Algorithm 5. Intuitively, we intend to use the offline algorithm as a subroutine to compute an allocation in each round independently. If we can prove that each separating point between two adjacent blocks is nondecreasing throughout the algorithm, then the $O(nT)$ upper bound is established. When the constraint factor in round t , defined as $B_t := \frac{1}{n} \cdot v(M_t) - \frac{n-1}{n} \cdot v^{\max}(M_t)$, is nondecreasing, we can show that all separating points are nondecreasing throughout the algorithm. Unfortunately, B_t may not be nondecreasing. For example, if $n = 3$ and there are two items g_1, g_2 such that $v(g_1) = 0$ and $v(g_2) = 1$, then we have $B_1 = 0$ and $B_2 = -1/3 < B_1$. Since in such cases where $B_{t+1} < B_t$, A^t satisfies the constraint factor B_t and adding the new item to the last block will only enable the allocation to be feasible for

a larger constraint factor, we just arbitrarily assign the new item to obtain A^{t+1} , which satisfies the constraint factor $B_t > B_{t+1}$. In fact, we will show that if $B_{t+1} < B_t^{\max}$ where $B_t^{\max} := \max_{i \in [t]} B_i$, then we can allocate the new item arbitrarily in this round.

In Algorithm 5, we use B_t to denote the proportional constraint factor for each block in round t . For any $i \in [N]$, p_i serves as the index of the last item in the i -th block, where $p_n = t$ is omitted. We update p_i sequentially by moving p_i backward until the i -th block satisfies the proportional constraint, i.e., $v(M_{p_{i-1}, p_i}) \geq B_t$ (Line 5–Line 7).

Algorithm 5: PROPa Algorithm for Identical Valuations

Input: valuation function v

```

1  $p_i \leftarrow 0$  for all  $i = 0, 1, 2, \dots, n-1$ 
2 for  $t = 1, \dots, T$  do
3    $B_t \leftarrow \frac{1}{n} \cdot v(M_t) - \frac{n-1}{n} \cdot v^{\max}(M_t)$ 
4   for  $i = 1, \dots, n-1$  do
5     while  $v(M_{p_{i-1}, p_i}) < B_t$  do
6        $p_i \leftarrow p_i + 1$ 
7     end
8      $A_i^t \leftarrow M_{p_{i-1}, p_i}$ 
9   end
10   $A_n^t \leftarrow M_{p_{n-1}, t}$ 
11 end
12 return  $[A^1, A^2, \dots, A^T]$ 

```

Theorem 4.2. *Algorithm 5 is a PROPa algorithm for identical valuations that requires $O(nT)$ adjustments.*

Proof. Denote $B_t^{\max} = \max_{i \in [t]} B_i$. For a contiguous allocation $A = (A_1, \dots, A_n)$, define $\ell(A_i)$ as the index of the last item in A_i . In particular, let $\ell(A_0) = 0$. We abuse the notation and let $\ell(A) = (\ell(A_1), \dots, \ell(A_{n-1}))$. For two contiguous allocations A' and A'' , we say that $\ell(A') \prec \ell(A'')$ if $\ell(A')$ is lexicographically smaller than $\ell(A'')$. We say that $\ell(A') \preceq \ell(A'')$ if $\ell(A') \prec \ell(A'')$ or $\ell(A') = \ell(A'')$. For any $t \in [T]$, define X^t as the allocation with lexicographically minimum $\ell(A)$ among all contiguous allocations $A = (A_1, \dots, A_n)$ for M_t satisfying constraint factor B_t^{\max} , i.e., $\min_{i \in N} v(A_i) \geq B_t^{\max}$. We first show that X^t is well-defined for all $t \in [T]$, and $\ell(X_i^t) \leq \ell(X_i^{t+1})$ for all $t < T$ and $i < n$. Then we prove that $A^t = X^t$ for all $t \in [T]$.

We first prove that X^t is well-defined for all $t \in [T]$, which is equivalent to showing the existence of the contiguous allocation for M_t that satisfies the constraint factor B_t^{\max} . The existence of X^t trivially holds for $t = 0$. We assume for induction that X^t exists and show that X^{t+1} also exists. In fact, if $B_{t+1}^{\max} = B_t^{\max}$, it is easy to show that by adding g_{t+1} to the last block of X^t , the corresponding allocation for M_{t+1} satisfies the constraint factor $B_{t+1}^{\max} = B_t^{\max}$ and we are done. On the other hand, if $B_{t+1}^{\max} > B_t^{\max}$, since in this case we must have $B_{t+1}^{\max} = B_{t+1}$, by Lemma 4.1 we know that the allocation that satisfies the constraint factor B_{t+1} exists. Therefore, X^t is well-defined for all $t \in [T]$.

Next, we show that for all $t < T$ and $i < n$, $\ell(X_i^t) \leq \ell(X_i^{t+1})$. Fix $t < T$. Suppose, for the sake of contradiction, that there exists $i < n$ such that $\ell(X_i^t) > \ell(X_i^{t+1})$ and $\ell(X_j^t) \leq \ell(X_j^{t+1})$ for all $j < i$. By the definition of $\ell(X_i^t)$ we know that $X_i^{t+1} \subsetneq X_i^t$. Since $B_1^{\max}, \dots, B_T^{\max}$ are nondecreasing, we have $v(X_i^{t+1}) \geq B_{t+1}^{\max} \geq B^t$ due to the definition of X_i^{t+1} . In allocation X^t , by moving the items in $X_i^t \setminus X_i^{t+1}$ from X_i^t to X_{i+1}^t , we obtain a new allocation X for M_t with $\ell(X) \prec \ell(X^t)$ that

satisfies the constraint factor B_t^{\max} , which contradicts the lexicographic minimality of X^t . Thus we conclude that $\ell(X_i^t) \leq \ell(X_i^{t+1})$ for all $t < T$ and $i < n$.

Now it suffices to prove that for all $t \in [T]$, $A^t = X^t$, which satisfies the constraint factor $B_t^{\max} \geq B_t$ by definition. First, $A^t = X^t$ trivially holds for $t = 0$. Supposing for induction that $A^{t-1} = X^{t-1}$ where $t < T$, we will show that $A^t = X^t$.

If $B_t^{\max} = B_{t-1}^{\max}$, then p_1, \dots, p_{n-1} will not change in round t since $A^{t-1} = X^{t-1}$ satisfies the constraint factor B_{t-1}^{\max} , which indicates that $\ell(A^t) = \ell(A^{t-1})$. By the lexicographic minimality of X^t , we have $\ell(X^t) \preceq \ell(A^t)$. Thus,

$$\ell(X^{t-1}) \preceq \ell(X^t) \preceq \ell(A^t) = \ell(A^{t-1}) = \ell(X^{t-1}),$$

by which we conclude that $A^t = X^t$.

Suppose $B_t^{\max} > B_{t-1}^{\max}$, which implies that $B_t = B_t^{\max}$. At the beginning of round t , we have

$$p_i = \ell(A_i^{t-1}) = \ell(X_i^{t-1}) \leq \ell(X_i^t),$$

for all $i < n$. It is trivial that $\ell(A_0^t) = \ell(X_0^t)$. Assuming for induction that $\ell(A_{i-1}^t) = \ell(X_{i-1}^t)$ where $i < n$, we will show that $\ell(A_i^t) = \ell(X_i^t)$. On one hand, $\ell(A_i^t) \leq \ell(X_i^t)$ must hold, since $v(X_i^t) \geq B_t$ and the condition of Line 5 will be violated when $p_i = \ell(X_i^t)$. On the other hand, $\ell(A_i^t) \geq \ell(X_i^t)$ must hold, since otherwise we have $A_i^t \subsetneq X_i^t$ and $v(A_i^t) \geq B_t$, and by moving the items in $X_i^t \setminus A_i^t$ from X_i^t to X_{i+1}^t , we obtain a new allocation X for M_t with $\ell(X) \prec \ell(X^t)$ that satisfies the constraint factor $B_t = B_t^{\max}$, which contradicts the lexicographic minimality of X^t . Therefore, we conclude that $A^t = X^t$ for all $t \in [T]$.

Finally, we prove the number of adjustments required by Algorithm 5. For an item g_t that has arrived, the agent i that g_t belongs to satisfies $p_{i-1} < t \leq p_i$. Since p_i is nondecreasing for each $i \in N$, the number of adjustments made on any item g_t is at most $n - 1$. As a result, the total number of adjustments is at most $(n - 1)T = O(nT)$. \square

4.2 Upper Bounds for EF1

In this section, we consider EF1 as the fairness criterion, which, as we will show later, is more strict than PROP. We first give an EF1 algorithm for two agents with general identical valuations that requires $O(T)$ adjustments in Algorithm 6. At each round t , we first find the minimum index i such that the total valuation of items on the left of g_i (inclusively) is at least the total valuation of the remaining items, i.e., $v(M_i) \geq v(M_{i,t})$ (Line 2). Then the line of items is separated into two blocks by g_i , and g_i is assigned to the block with the smaller valuation (Line 3–Line 7). In fact, such i specified above is called *lumpy tie* in [9], which possesses some desirable properties that are useful to establish contiguous fair allocations.

Theorem 4.3. *Algorithm 6 is an EF1 algorithm for two agents with general identical valuations that requires $O(T)$ adjustments.*

Proof. First, the i defined in Line 2 is well-defined since $t \in \{j \mid v(M_j) \geq v(M_{j,t})\}$. Then we show that A^t is EF1 for any $t \in [T]$. Fix $t \in [T]$ and define i as in Line 2. Let $L = M_{i-1}$ and $R = M_{i,t}$, and we have $v(L + g_i) \geq v(R)$ by definition. Note that $v(R + g_i) > v(L)$ holds since otherwise $v(L) \geq v(R + g_i)$ and we know that $i - 1 \in \{j \mid v(M_j) \geq v(M_{j,t})\}$, which contradicts the minimality of i . By symmetry, suppose that $A^t = (L \cup \{g_i\}, R)$, which means that $v(L) \leq v(R)$. Thus agent 2 does not envy agent 1 up to one item. On the other hand, we know that $v(L + g_i) \geq v(R)$ and thus agent 1 does not envy agent 2. As a result, we conclude that A^t is EF1 for any $t \in [T]$.

Algorithm 6: EF1 Algorithm for Two Agents with General Valuations

Input: valuation function v

```

1 for  $t = 1, \dots, T$  do
2    $i = \min\{j \mid v(M_j) \geq v(M_{j,t})\}$ 
3   if  $v(M_{i-1}) \leq v(M_{i,t})$  then
4      $A^t = (M_i, M_{i,t})$ 
5   else
6      $A^t = (M_{i-1}, M_{i-1,t})$ 
7   end
8 end
9 return  $[A^1, A^2, \dots, A^T]$ 

```

It suffices to prove the required number of adjustments. Define $f(t) = \min\{j \mid v(M_j) \geq v(M_{j,t})\}$ and $h(t)$ such that agent 1 obtains exactly the first $h(t)$ items in A^t . By the monotonicity of v , we know that $f(t)$ is nondecreasing. Moreover, it is easy to see that $f(t) = h(t)$ or $f(t) = h(t) - 1$. It suffices to prove that $h(t)$ is nondecreasing since in this case, the total number of adjustments is

$$\sum_{t=1}^{T-1} |h(t+1) - h(t)| = \sum_{t=1}^{T-1} (h(t+1) - h(t)) = h(T) - h(1) \leq T.$$

In fact, fixing $t < T$, if $f(t) < f(t+1)$, then it holds that

$$h(t) \leq f(t) \leq f(t+1) - 1 \leq h(t+1).$$

On the other hand, if $f(t) = f(t+1)$ and $h(t) = f(t) - 1$, then $h(t) \leq h(t+1)$ trivially holds. Otherwise, if $f(t) = f(t+1)$ and $h(t) = f(t)$, then we have $v(M_{f(t)-1}) \leq v(M_{f(t),t})$. Since

$$v(M_{f(t+1)-1}) = v(M_{f(t)-1}) \leq v(M_{f(t),t}) \leq v(M_{f(t),t+1}) = v(M_{f(t+1),t+1}),$$

we know that $h(t+1) = f(t+1) = h(t)$. Therefore, $h(t)$ is nondecreasing.

All the above concludes the proof. \square

For any number of agents, we fail to establish such a nondecreasing property for the separating points, by which we can similarly achieve $O(nT)$ adjustments. Under the assumption that the valuation of each item lies in some range $[L, R]$ such that $L > 0$, we employ the slightly modified version of the algorithm given by [9] for any number of agents with additive identical valuations as a subroutine to obtain an EF1 allocation in each round independently. It immediately leads to an EF1 algorithm for additive identical valuations that requires $O(\frac{R}{L} \cdot n^2 T)$ adjustments.

We present the offline algorithm in Algorithm 7 and the online algorithm in Algorithm 8. For a contiguous allocation A , define $P(A) = (\ell_0, \dots, \ell_n)$ such that $\ell_0 = 0$ and $P_i(A) = \ell_i$ is the index of the last item in A_i . Denote the *leximin* (*lexicographic maximin*) allocation as the allocation that maximizes the lowest valuation among all agents, subject to which it maximizes the second-lowest one, and so on. We say that A is *leximin*² if A is the *leximin* allocation with lexicographically smallest $P(A)$ among all *leximin* allocations. In Algorithm 7, we first find a contiguous *leximin*² allocation A as the initial allocation and fix an agent i with the minimum valuation in the initial allocation. Then we iterate from 1 to $i-1$ and from n to $i+1$ separately. During each iteration j , whenever agent i envies agent j even up to one item, the item in A_j that is closest to A_i is moved to the adjacent block. The only difference between Algorithm 7 and the algorithm given by [9]

is that Algorithm 7 uses the leximin² allocation as the initial allocation rather than an arbitrary leximin allocation. Intuitively, the leximin² allocations possess the property that each separating point between two adjacent blocks is nondecreasing, which is crucial to prove the upper bound. For completeness, we show how to use Dynamic Programming to efficiently compute the contiguous leximin² allocation in Appendix B.

Algorithm 7: EF1 Offline Algorithm

Input: valuation function v , item set M

- 1 Let $A = (A_1, \dots, A_n)$ be the contiguous leximin² allocation of M
- 2 Fix $i = \arg \min_{j \in [N]} v(A_j)$
- 3 **for** $j = 1, \dots, i - 1$ **do**
- 4 **while** agent i envies agent j even up to one item **do**
- 5 | move the rightmost item of A_j to A_{j+1}
- 6 **end**
- 7 **end**
- 8 **for** $j = n, \dots, i + 1$ **do**
- 9 **while** agent i envies agent j even up to one item **do**
- 10 | move the leftmost item of A_j to A_{j-1}
- 11 **end**
- 12 **end**
- 13 **return** A

Algorithm 8: EF1 Online Algorithm

Input: valuation function v

- 1 **for** $t = 1, \dots, T$ **do**
- 2 | $A^t \leftarrow$ the allocation returned by Algorithm 7 running with M_t
- 3 **end**
- 4 **return** $[A^1, A^2, \dots, A^T]$

We first present the characterization proved by [9] of the offline algorithm.

Lemma 4.4 ([9]). *For additive identical valuations, Algorithm 7 is EF1 and can be implemented in polynomial time. Moreover, throughout the algorithm, A_i does not change, where i is the agent fixed at the beginning of Algorithm 7.*

Now we formally show the performance of the online algorithm.

Theorem 4.5. *Suppose that the valuations are additive identical and there exist $0 < L \leq R$ such that $v(g) \in [L, R]$ for any $g \in M$. Then Algorithm 8 is EF1 and requires $O\left(\frac{R}{L} \cdot n^2 T\right)$ adjustments.*

Before proving Theorem 4.5, we give some useful lemmas to characterize the initial contiguous leximin² allocation of Algorithm 7 in each round. Lemma 4.6 provides an upper bound for the valuation of each block in a leximin allocation, and Lemma 4.7 proves the monotonicity of the separating points in the leximin² allocations. The proof of Lemma 4.7 is deferred to Appendix C.

Lemma 4.6. *For additive identical valuations, suppose that the valuation of each item is upper bounded by R . Let A be a leximin contiguous allocation of M and i satisfy $v(A_i) \leq v(A_j)$ for all $j \in N$. Let $B = v(A_i)$. Then throughout Algorithm 7, $v(A_j) \leq |i - j|R + B$ always holds for any $j \in N$.*

Proof. Due to the symmetry, we assume that $i = 1$ and we will prove that

$$v(A_j) \leq (j - 1)R + B \quad (3)$$

always holds for any $j \in N$. Note that by Lemma 4.4, A_i does not change throughout the algorithm. Thus (3) always holds for $j = 1$. Now supposing for induction that (3) always holds for $j = k - 1$ where $k > 1$, we will show that (3) also holds for $j = k$ all the time. For the sake of contradiction, suppose that $v(A_k) > (k - 1)R + B$ at some time. When the iteration variable becomes k , we will keep moving the leftmost item in A_k to A_{k-1} until agent i does not envy agent k up to one item. Note that agent i does not envy agent k up to one item only if $v(A_k) \leq v(A_i) + R = B + R$, since otherwise for any $g \in A_k$, $v(A_k - g) \geq v(A_k) - R > v(A_i)$. Thus the total valuation of the items being moved from A_k to A_{k-1} during the iteration is at least

$$v(A_k) - (v(A_i) + R) > (k - 1)R + B - (B + R) = (k - 2)R,$$

which, combining the fact that $v(A_{k-1}) \geq B$, contradicts the induction assumption that (3) always holds for $j = k - 1$. \square

Lemma 4.7. *For additive identical valuations, let X^t be the initial contiguous leximin² allocation fixed by Algorithm 7 in round t . Then for every $t < T$ and $j \in N$, we have $P_j(X^t) \leq P_j(X^{t+1})$.*

Lemma 4.7 implies the existence of a polynomial-time algorithm that requires $O(nT)$ adjustments to achieve leximin, which is a powerful tool with many applications. Now it is sufficient to prove Theorem 4.5.

Proof of Theorem 4.5. By Lemma 4.4 we know that Algorithm 8 is EF1. It suffices to prove the required number of adjustments. Let X^t be the initial leximin² allocation and i^t be the fixed agent with the minimum valuation in X^t in round t . We will use Lemma 4.6 to show that for any $t \in [T]$, the number of adjustments needed to transform X^t to A^t is upper bounded by $O(n^2R/L)$. Combining the monotonicity of $P_j(X^t)$ for all $j \in N$ given by Lemma 4.7, we can establish the desired upper bound by triangle inequality.

Fix round t . By Lemma 4.6, we know that $v(A_j) \leq v(A_i) + |i - j|R$ throughout Algorithm 7. Since by Lemma 4.4, A_i never changes and the valuation of each item is lower bounded by L , the number of items being moved from A_j toward A_i during the iteration j is at most $(v(A_j) - v(A_i))/L \leq |i - j|R/L$. Note that for two contiguous allocations A' and A'' , the number of adjustments that are required to transform A' to A'' is $\sum_{j=1}^{n-1} |P_j(A') - P_j(A'')|$. Consequently, the number of adjustments required by Algorithm 7 to transform the initial contiguous leximin² allocation X^t to the final EF1 allocation A^t is upper bounded by

$$\sum_{j=1}^{n-1} |P_j(X^t) - P_j(A^t)| \leq \sum_{j \neq i} |i - j|R/L \leq O(n^2R/L).$$

Therefore, the total number of adjustments required by Algorithm 8 is

$$\sum_{t=1}^{T-1} \sum_{j=1}^{n-1} |P_j(A^t) - P_j(A^{t+1})| \quad (4)$$

$$= \sum_{t=1}^{T-1} \sum_{j=1}^{n-1} (|P_j(A^t) - P_j(X^t)| + |P_j(X^t) - P_j(X^{t+1})| + |P_j(X^{t+1}) - P_j(A^{t+1})|) \quad (5)$$

$$\leq \sum_{t=1}^{T-1} \sum_{j=1}^{n-1} |P_j(X^t) - P_j(X^{t+1})| + O\left(\frac{R}{L} \cdot n^2 T\right) \quad (6)$$

$$= \sum_{t=1}^{T-1} \sum_{j=1}^{n-1} (P_j(X^{t+1}) - P_j(X^t)) + O\left(\frac{R}{L} \cdot n^2 T\right) \quad (7)$$

$$= \sum_{j=1}^{n-1} (P_j(X^T) - P_j(X^1)) + O\left(\frac{R}{L} \cdot n^2 T\right) \quad (8)$$

$$\leq O(nT) + O\left(\frac{R}{L} \cdot n^2 T\right) = O\left(\frac{R}{L} \cdot n^2 T\right), \quad (9)$$

where (7) is due to the monotonicity of $P_j(X^t)$ for any $j \in N$ given by Lemma 4.7. \square

4.3 Lower Bounds

We demonstrate that our upper bound to achieve PROP_a for identical valuations is asymptotically tight. Moreover, by showing that EF1 can imply PROP_a, the same lower bound is also applied to EF1.

Theorem 4.8. *For identical valuations, every PROP_a algorithm requires at least $\Omega(nT)$ adjustments.*

Proof. Suppose that there are n agents with identical valuations and each agent has a valuation of 1 for every item. The proportional constraint factor for M_T that should be satisfied by each block is

$$\left\lceil \frac{T}{n} - \frac{n-1}{n} \right\rceil = \left\lfloor \frac{T}{n} \right\rfloor.$$

Denote the number of items that have arrived as T . We first prove that, for a large enough t , g_t must come to every agent's block at least once as T grows larger. In other words, for every agent i , there exists T_i such that in round T_i , g_t belongs to agent i . Then we use this property to show the desired lower bound.

We will prove that, for $t \geq n^2$, g_t must come to every agent's block at least once as T grows larger. Fix $t \geq n^2$. Note that in round $T = t$, g_t must belong to agent n , since agent n 's block cannot be empty. Now we show that for any $i < n$, agent i will obtain g_t at some time. Suppose $t = ki - r$, where $0 \leq r < i$ and $k \geq n$. In round $T = kn > t$, each agent should obtain exactly k items. Then g_t belongs to agent i since $k(i-1) < t \leq ki$.

Now it is sufficient to give the lower bound. By the above arguments, for any $t \geq n^2$, $T \geq t$ and $i \geq nt/(T-n)$, we know that g_t must have been obtained by agents $i, i+1, \dots, n$ at some time.

Thus the number of adjustments made on g_t throughout T rounds is at least $n - \lceil nt/(T - n) \rceil$. Summing over all $t \geq n^2$, we have

$$\begin{aligned} \sum_{t=n^2}^T \left(n - \left\lceil \frac{nt}{T-n} \right\rceil \right) &\geq \sum_{t=n^2}^T \left(n - \frac{nt}{T-n} - 1 \right) \geq (n-1)(T-n^2) - \sum_{t=1}^T \frac{nt}{T-n} \\ &= (n-1)(T-n^2) - \frac{nT(T+1)}{2(T-n)} = \Omega(nT), \end{aligned}$$

where the last equality is due to $T \gg n$.

Therefore, we conclude that the number of adjustments required by any PROP algorithm is at least $\Omega(nT)$. \square

It is worth noting that the proportionality condition (2) can be implied by EF1 even for non-identical valuations. Thus the $O(nT)$ lower bound also works for EF1.

Lemma 4.9. *For agents with nonidentical valuations, if an allocation is EF1, then it also satisfies (2) for all $i \in N$.*

Proof. Suppose that there are n agents with valuation functions v_1, \dots, v_n and allocation $A = (A_1, \dots, A_n)$ is EF1. Fix $i \in N$. By the definition of EF1, for any $j \in N$, there exists $g_j \in A_j$ such that

$$v_i(A_i) \geq v_i(A_j - g_j).$$

Summing over all $j \neq i$, we obtain

$$\begin{aligned} (n-1)v_i(A_i) &\geq \sum_{j \neq i} v_i(A_j - g_j) \\ \implies nv_i(A_i) &\geq v_i(M) - \sum_{j \neq i} v_i(g_j) \\ \implies v_i(A_i) &\geq \frac{1}{n} \cdot v_i(M) - \frac{1}{n} \sum_{j \neq i} v_i(g_j) \geq \frac{1}{n} \cdot v_i(M) - \frac{n-1}{n} \cdot v_i^{\max}(M) \end{aligned}$$

as desired. \square

Corollary 4.10. *For identical valuations, every EF1 algorithm requires at least $\Omega(nT)$ adjustments.*

5 Contiguous Setting with Nonidentical Valuations

For nonidentical valuations, unfortunately, we give an instance to show that it is impossible to make any significant improvement to the trivial $O(T^2)$ upper bound for PROP.

Theorem 5.1. *For nonidentical valuations, every PROP algorithm requires at least $\Omega(T^2/n)$ adjustments.*

Proof. Suppose that there are T items and n agents with valuation functions v_1, \dots, v_n . For any $t \in [T]$, let

$$v_i(g_t) = \begin{cases} n^{2c}, & cn^2 + (i-1)n < t \leq cn^2 + in \text{ for some } c \geq 0, \\ 0, & \text{otherwise,} \end{cases}$$

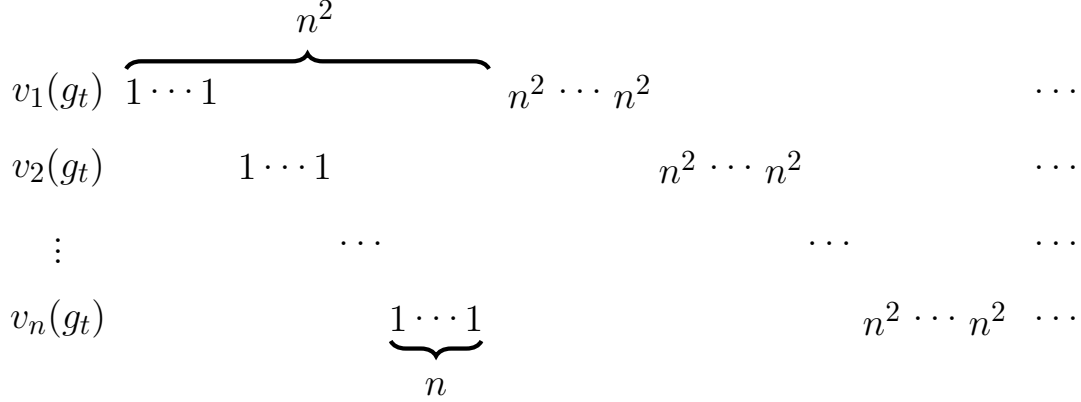


Figure 1: Figure illustrating the instance in the proof of Theorem 5.1. Each period of length n^2 is divided into n blocks of length n , and agent i is only interested in the items in the i -th block in each period c with n^{2c} valuation for each of these items. If a valuation is 0, then it is omitted.

for all $i \in N$. As illustrated in Figure 1, each period of length n^2 is divided into n blocks of length n , and agent i is only interested in the items in the i -th block in each period c with n^{2c} valuation for each of these items. It suffices to show that, for every $k \geq 2n$, the first $nk - n^2$ items belong to one agent in round nk and belong to a different agent in round $nk + n$. Since in this case, if we assume that $T \gg n$, the number of adjustments required is at least $\sum_{k=2n}^{T/n-1} (nk - n^2) = \Theta(T^2/n)$.

We only give the proof for k such that k is a multiple of n , and the proof can be easily generalized to any $k \geq 2n$. Now we prove that, for any $c \geq 2$ and $k = cn$, the first $nk - n^2 = (c-1)n^2$ items belong to agent 1 in round nk and belong to agent 2 in round $nk + n$. Note that in round nk , the proportional constraint factor for all agents is

$$\frac{1}{n} \left(n \sum_{j=0}^{c-1} n^{2j} \right) - \frac{n-1}{n} \cdot n^{2(c-1)} = n^{2c-3} + \sum_{j=0}^{c-2} n^{2j},$$

and the total valuation of the first $nk - n^2$ items for each agent is

$$n \sum_{j=0}^{c-2} n^{2j} = \frac{n^{2(c-1)} - 1}{n+1} + \sum_{j=0}^{c-2} n^{2j} < n^{2c-3} + \sum_{j=0}^{c-2} n^{2j}.$$

Define $G_i = \{g_t \mid (c-1)n^2 + (i-1)n < t \leq (c-1)n^2 + in\}$ as the set of items that arrive during period c and agent i is interested in. Thus to satisfy the proportional constraint, each agent i should get at least one of the items in G_i . Due to the contiguity requirement, agent i must get the i -th block which should contain at least one item in G_i . In this case, the first $nk - n^2 = (c-1)n^2$ items belong to agent 1. Similarly, we can show that, in round $nk + n$, the first $nk - n^2 + n$ items belong to agent 2 and we are done.

Therefore, we conclude that any PROP algorithm requires at least $\Omega(T^2/n)$ adjustments. \square

Next, we show that when it comes to EF1, the lower bound is even stronger, indicating that we cannot make any improvement. The proof of Corollary 5.2 is analogous to the proof of Theorem 5.1, and can be found in Appendix D.

Theorem 5.2. *For nonidentical valuations, every EF1 algorithm requires at least $\Omega(T^2)$ adjustments.*

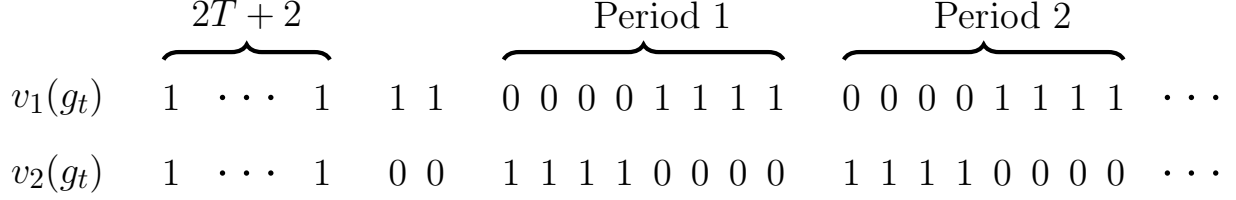


Figure 2: Figure illustrating the instance in the proof of Theorem 5.3

5.1 Lower Bound for Binary Valuations

Note that in the hard instance given to prove the lower bound for nonidentical valuations, i.e., Theorem 5.1, the valuations of items are unbounded. We show that even for 2 agents with binary valuations, it is also impossible to improve the trivial $O(T^2)$ upper bound. Note that for 2 agents with binary valuations, PROP is equivalent to EF1.

Theorem 5.3. *For 2 agents with binary valuations, every EF1 algorithm requires at least $\Omega(T^2)$ adjustments.*

Proof. We describe an instance with $4T + 4$ items where $T > 0$ is a multiple of 4. The instance is illustrated in Figure 2. We say that an item g is type 0 if $v_1(g) = v_2(g) = 1$, type 1 if $v_1(g) = 1, v_2(g) = 0$, and type 2 if $v_1(g) = 0, v_2(g) = 1$. The first $2T + 2$ items are type 0. The following 2 items are type 1. The remaining $2T$ items are divided into periods with a length of 8. In each period, the first 4 items are type 2 and the last 4 items are type 1. For any $0 \leq k < T/4$, we will show that the first block must belong to agent 2 in round $2T + 8k + 4$ and must belong to agent 1 in round $2T + 8k + 8$.

Given an allocation, we say that an item is non-wasteful if it has a valuation of 1 for the agent that obtains it. Notice that in round $2T + 8k + 4$, the numbers of non-wasteful items required by agent 1 and agent 2 are $T + 2k + 2$ and $T + 2k + 1$, respectively. Thus an EF1 allocation should contain at least $2T + 4k + 3$ non-wasteful items. Suppose, for the sake of contradiction, that the second block belongs to agent 2. Denote G as the set of the last $8k + 2$ items. Since $v_2(G) = 4k < T + 2k + 1$, all items in G must belong to agent 2, and thus there are $|G| - v_2(G) = 4k + 2$ wasteful items in G . As a result, the number of non-wasteful items is at most

$$(2T + 8k + 4) - (4k + 2) = 4T + 4k + 2 < 2T + 4k + 3,$$

which leads to a contradiction. Therefore, we conclude that the first block must belong to agent 2 in round $2T + 8k + 4$.

The proof for round $2T + 8k + 8$ is analogous. In round $2T + 8k + 8$, the numbers of non-wasteful items required by agent 1 and agent 2 are $T + 2k + 2$ and $T + 2k + 3$, respectively. Thus an EF1 allocation should contain at least $2T + 4k + 5$ non-wasteful items. Suppose, for the sake of contradiction, that the second block belongs to agent 1. Denote G as the set of the last $8k + 6$ items. Since $v_1(G) = 4k + 2 < T + 2k + 3$, all items in G must belong to agent 1, and thus there are $|G| - v_1(G) = 4k + 4$ wasteful items in G . As a result, the number of non-wasteful items is at most

$$(2T + 8k + 8) - (4k + 4) = 2T + 4k + 4 < 2T + 4k + 5,$$

which leads to a contradiction. Therefore, we conclude that the first block must belong to agent 1 in round $2T + 8k + 8$.

Finally, we prove the lower bound of the number of required adjustments. Starting from round $2T + 4$, the agent who obtains the first block alternates after every 4 rounds. Since the first block must contain the first T items, the number of adjustments is at least $T^2/2 = \Theta(T^2)$. \square

Unfortunately, the instance given in the proof of the lower bound for two agents with binary valuations only provides a $\Omega(nT)$ lower bound after being generalized to any number of agents, which is directly implied by Theorem 4.8.

6 Conclusion and Future Work

In this work, we study the problem of dynamically allocating indivisible items in a fair manner with few adjustments in both the contiguous setting and the noncontiguous setting. We present some directions for future work.

- Even though we establish tight upper and lower bounds to achieve PROP_a in the contiguous setting, there are large gaps between the upper and lower bounds to achieve EF1 in both settings. The first open problem is to tighten these bounds.
- If the types of the items are drawn from certain distributions rather than adversely, can we show a better upper bound in expectation or asymptotically in both settings? Similar problems are presented in [28].
- It would be interesting to investigate other fairness notions like EQ1 [25].
- With the contiguity constraint, we showed that it is impossible to make each agent better off if we fix the order of agents. When the order of agents is allowed to change, can we make any progress if the incentive property must hold?
- Another promising direction, like always asked in the offline setting [16], is to achieve fairness and Pareto optimality simultaneously. This has been shown in some other online models [3, 29].

Acknowledgment

We would like to thank Alexandros Psomas for the discussion about the content in the early stage of this work and his helpful suggestion on the presentation. We are also grateful to Ruta Mehta for inspiring us to think about the incentive issue in this model.

References

- [1] Hannaneh Akrami, Rojin Rezvan, and Masoud Seddighin. An EF2X allocation protocol for restricted additive valuations. In *IJCAI*, pages 17–23. ijcai.org, 2022.
- [2] Martin Aleksandrov and Toby Walsh. Expected outcomes and manipulations in online fair division. In *KI*, volume 10505 of *Lecture Notes in Computer Science*, pages 29–43. Springer, 2017.
- [3] Martin Aleksandrov and Toby Walsh. Strategy-proofness, envy-freeness and pareto efficiency in online fair division with additive utilities. In *PRICAI (1)*, volume 11670 of *Lecture Notes in Computer Science*, pages 527–541. Springer, 2019.

- [4] Martin Aleksandrov and Toby Walsh. Online fair division: A survey. In *AAAI*, pages 13557–13562. AAAI Press, 2020.
- [5] Haris Aziz, Ioannis Caragiannis, Ayumi Igarashi, and Toby Walsh. Fair allocation of indivisible goods and chores. *Auton. Agents Multi Agent Syst.*, 36(1):3, 2022.
- [6] Yushi Bai and Paul Gözl. Envy-free and pareto-optimal allocations for agents with asymmetric random valuations. In *IJCAI*, pages 53–59. ijcai.org, 2022.
- [7] Gerdus Benadè, Daniel Halpern, and Alexandros Psomas. Dynamic fair division with partial information.
- [8] Gerdus Benade, Aleksandr M. Kazachkov, Ariel D. Procaccia, and Christos-Alexandros Psomas. How to make envy vanish over time. In *EC*, pages 593–610. ACM, 2018.
- [9] Vittorio Bilò, Ioannis Caragiannis, Michele Flammini, Ayumi Igarashi, Gianpiero Monaco, Dominik Peters, Cosimo Vinci, and William S. Zwicker. Almost envy-free allocations with connected bundles. *Games Econ. Behav.*, 131:197–221, 2022.
- [10] Sylvain Bouveret, Katarína Cechlárová, Edith Elkind, Ayumi Igarashi, and Dominik Peters. Fair division of a graph. In *IJCAI*, pages 135–141. ijcai.org, 2017.
- [11] Sylvain Bouveret, Katarína Cechlárová, and Julien Lesca. Chore division on a graph. *Auton. Agents Multi Agent Syst.*, 33(5):540–563, 2019.
- [12] Johannes Brustle, Jack Dippel, Vishnu V. Narayan, Mashbat Suzuki, and Adrian Vetta. One dollar each eliminates envy. In *EC*, pages 23–39. ACM, 2020.
- [13] Eric Budish. The combinatorial assignment problem: approximate competitive equilibrium from equal incomes. In *BQGT*, page 74:1. ACM, 2010.
- [14] Eric J. Friedman, Christos-Alexandros Psomas, and Shai Vardi. Dynamic fair division with minimal disruptions. In *EC*, pages 697–713. ACM, 2015.
- [15] Eric J. Friedman, Christos-Alexandros Psomas, and Shai Vardi. Controlled dynamic fair division. In *EC*, pages 461–478. ACM, 2017.
- [16] Jugal Garg and Aniket Murhekar. Computing pareto-optimal and almost envy-free allocations of indivisible goods. *CoRR*, abs/2204.14229, 2022.
- [17] Jonathan R. Goldman and Ariel D. Procaccia. Spliddit: unleashing fair division algorithms. *SIGecom Exch.*, 13(2):41–46, 2014.
- [18] Jiafan He, Ariel D. Procaccia, Alexandros Psomas, and David Zeng. Achieving a fairer future by changing the past. In *IJCAI*, pages 343–349. ijcai.org, 2019.
- [19] Felix Höhne and Rob van Stee. Allocating contiguous blocks of indivisible chores fairly. *Inf. Comput.*, 281:104739, 2021.
- [20] Ayumi Igarashi. How to cut a discrete cake fairly, 2022.
- [21] Richard J. Lipton, Evangelos Markakis, Elchanan Mossel, and Amin Saberi. On approximately fair allocations of indivisible goods. In *EC*, pages 125–131. ACM, 2004.

- [22] Vasilis Livanos, Ruta Mehta, and Aniket Murhekar. (almost) envy-free, proportional and efficient allocations of an indivisible mixed manna. In *AAMAS*, pages 1678–1680. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2022.
- [23] Pasin Manurangsi and Warut Suksompong. When do envy-free allocations exist? *SIAM J. Discret. Math.*, 34(3):1505–1521, 2020.
- [24] Pasin Manurangsi and Warut Suksompong. Closing gaps in asymptotic fair division. *SIAM J. Discret. Math.*, 35(2):668–706, 2021.
- [25] Neeldhara Misra, Chinmay Sonar, P. R. Vaidyanathan, and Rohit Vaish. Equitable division of a path. *CoRR*, abs/2101.09794, 2021.
- [26] Hoon Oh, Ariel D. Procaccia, and Warut Suksompong. Fairly allocating many goods with few queries. *SIAM J. Discret. Math.*, 35(2):788–813, 2021.
- [27] Roman Seidl. Handbook of computational social choice *by Brandt Felix, Vincent Conitzer, Ulle Endriss, Jerome Lang, Ariel Procaccia. J. Artif. Soc. Soc. Simul.*, 21(2), 2018.
- [28] Warut Suksompong. Fairly allocating contiguous blocks of indivisible items. *Discret. Appl. Math.*, 260:227–236, 2019.
- [29] David Zeng and Alexandros Psomas. Fairness-efficiency tradeoffs in dynamic fair division. In *EC*, pages 911–912. ACM, 2020.

A Incentive Property of Round-Robin Algorithm

Recall that in the round-robin algorithm, agents pick their favorite items alternately in each round. We will show that if we re-run the round-robin algorithm with a deterministic tie-breaking rule upon the arrival of a new good, then the valuation of each agent will not decrease. This immediately leads to the incentive property of the algorithms given by [18] which are based on the round-robin algorithm.

Suppose that there are n agents and m items g_1, \dots, g_m . In the outcome of the round-robin algorithm, we use C_i^j to denote the index of the item obtained by agent i in round j . In particular, $C_i^j = 0$ if agent i does not obtain any item in round j , and let $v_i(0) = 0$ for all $i \in N$. Note that there are $\lceil m/n \rceil$ rounds in total. When a new item g_{m+1} arrives and we run the round-robin algorithm again to assign all the $m+1$ items to agents, let \tilde{C}_a^i denote the outcome analogously. Now we formally present the incentive property in the following lemma.

Lemma A.1. *For every $1 \leq j \leq \lceil (m+1)/n \rceil$ and $i \in N$, we have $v_i(C_i^j) \leq v_i(\tilde{C}_i^j)$.*

Proof. Without loss of generality, we assume that the agent with a smaller index always picks an item before the agent with a larger index in each round. Define

$$M_i^j = \{g_1, \dots, g_m\} \setminus \{C_k^\ell \mid \text{either } \ell < j \text{ or } \ell = j, k < i\}$$

as the set of remaining items before agent i pick an item in round j , and

$$\tilde{M}_i^j = \{g_1, \dots, g_{m+1}\} \setminus \{\tilde{C}_k^\ell \mid \text{either } \ell < j \text{ or } \ell = j, k < i\}$$

analogously. Since

$$C_i^j = \arg \max_{g \in M_i^j} v_i(g), \quad \tilde{C}_i^j = \arg \max_{g \in \tilde{M}_i^j} v_i(g),$$

it suffices to show that $M_i^j \subseteq \tilde{M}_i^j$, which trivially holds when $j = 0$.

Assume that $i < n$, and the case of $i = n$ is treated similarly. Suppose for induction that $M_i^j \subseteq \tilde{M}_i^j$ is satisfied, we show that $M_{i+1}^j \subseteq \tilde{M}_{i+1}^j$. Noting that $|M_i^j| + 1 = |\tilde{M}_i^j|$, let $g \in \tilde{M}_i^j \setminus M_i^j$. Now there are two possible cases. On one hand, if $\tilde{C}_i^j = g$, then $\tilde{M}_{i+1}^j = M_{i+1}^j \cup \{C_i^j\}$. On the other hand, if $\tilde{C}_i^j \neq g$, then we have $\tilde{C}_i^j = C_i^j$ and thus $\tilde{M}_{i+1}^j = M_{i+1}^j \cup \{g\}$. The proof is completed by induction. \square

B Efficient Computation of Contiguous leximin² Allocations for Identical Valuations

In this section, we discuss how to efficiently compute contiguous leximin² allocations to serve as the initial allocation of Algorithm 7. Given an instance with n agents and m items, we will apply Dynamic Programming to compute the contiguous leximin² allocation that runs in time $O(n^2 m^2)$.

Suppose that there are n agents with general identical valuations v_1, \dots, v_n and m items g_1, \dots, g_m arranged in a line. Recall that $M_{l,r} = \{g_{l+1}, \dots, g_r\}$ is the set of items whose indexes are in range $[l+1, r]$. In particular, let $M_{l,r} = \emptyset$ if $l \geq r$. Recall that the leximin² allocation is the leximin allocation A with lexicographically minimum $P(A)$ among all leximin allocations, where $P(A) = (\ell_0, \dots, \ell_n)$ such that $\ell_0 = 0$ and ℓ_i is the index of the last item in A_i . For an allocation $A = (A_1, \dots, A_k)$, define $Q(A) = (v(A_{\sigma(1)}), v(A_{\sigma(2)}), \dots, v(A_{\sigma(k)}))$ as the tuple obtained by sorting

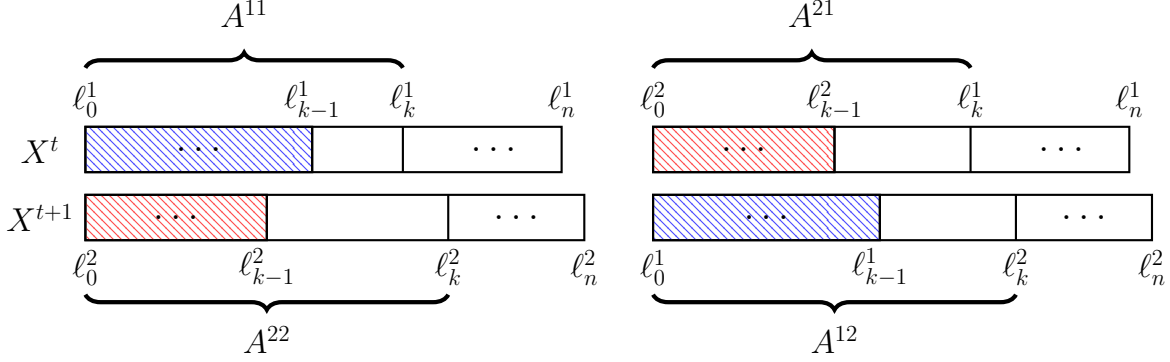


Figure 3: Figure illustrating the proof of Lemma 4.7

the valuations of all agents in nondecreasing order, where σ is a permutation with length k such that $v(A_{\sigma(i)}) \leq v(A_{\sigma(i+1)})$ for any $i < k$. Given two different allocations A' and A'' , we say that A' is better than A'' if either $Q(A')$ is lexicographically larger than $Q(A'')$ or $Q(A') = Q(A'')$ and $P(A')$ is lexicographically smaller than $P(A'')$. By definition, the leximin² allocation is the best allocation among all contiguous allocations.

For any $i \in [n]$ and $j \in [m] \cup \{0\}$, let $A[i, j]$ be the leximin² allocation for M_j with i agents. It is easy to observe that if $A = (A_1, \dots, A_i)$ is a leximin² allocation, then for any $k \in [i]$, $A' = (A_1, \dots, A_k)$ is the leximin² allocation for the first $|A_1| + \dots + |A_k|$ items with k agents. Thus if the last block of $A[i, j]$ is $M_{k,j}$ for some $0 \leq k \leq j$, then $A[i, j] = A[i-1, k] \circ M_{k,j}$, where we use $A \circ x$ to denote the tuple obtained by appending x to the end of tuple A . As a result, the recurrence relation of $A[i, j]$ is

$$A[i, j] = \begin{cases} (M_j), & i = 1, \\ \text{the best allocation in } \{A[i-1, k] \circ M_{k,j} \mid 0 \leq k \leq j\}, & \text{otherwise.} \end{cases}$$

When it comes to implementation, we only need to maintain $P(A[i, j])$ and $Q(A[i, j])$, which are the necessary information to compare two allocations, rather than recording the exact allocation. We use $f[i, j]$ to denote $P(A[i, j])$ and $h[i, j]$ to denote $Q(A[i, j])$. By the recurrence relation of $A[i, j]$, we have $f[i, j] = f[i-1, k] \circ j$, where the last block of $A[i, j]$ is $M_{k,j}$. Analogously, $h[i, j]$ can be derived by inserting $v(M_{k,j})$ to the appropriate position in $h[i-1, k]$ to ensure the monotonicity of the elements in $h[i, j]$.

Now we analyze the time complexity of the Dynamic Programming approach that we discussed above. For each pair $(i, j) \in [n] \times ([m] \cup \{0\})$, we need to enumerate all allocations in $\{A[i-1, k] \circ M_{k,j} \mid 0 \leq k \leq j\}$ and make comparisons among them to compute $A[i, j]$. Thus the total number of comparisons to be made is $O(nm^2)$. Moreover, each comparison can be finished in time $O(n)$. Therefore, the running time of the algorithm is $O(n^2m^2)$.

C Proof of Lemma 4.7

Let's first specify some conventions. Since for any feasible tuple $O = (\ell_0, \dots, \ell_n)$, there only exists one allocation A satisfying $P(A) = O$, from now on we will use a tuple to refer to the corresponding allocation. For an allocation $A = (A_1, \dots, A_k)$ and $O = P(A)$, define $Q(O) = (v(A_{\sigma(1)}), v(A_{\sigma(2)}), \dots, v(A_{\sigma(k)}))$ as the tuple obtained by sorting the valuations of all agents in nondecreasing order, where σ is a permutation with length k such that $v(A_{\sigma(i)}) \leq v(A_{\sigma(i+1)})$ for

any $i < k$. For two tuples O_1 and O_2 , we say $O_1 \preceq O_2$ if O_1 is lexicographically not larger than O_2 . Without loss of generality, we assume that $v(g) > 0$ for all $g \in M$.

Fix round t . We will prove that $P_j(X^t) \leq P_j(X^{t+1})$ for every $j \in N$. Let $P_j(X^t) = (\ell_0^1, \ell_1^1, \dots, \ell_n^1)$ and $P_j(X^{t+1}) = (\ell_0^2, \ell_1^2, \dots, \ell_n^2)$, as illustrated in Figure 3. Since both X^t and X^{t+1} are leximin², for any $j \in [n]$ and $k \in \{1, 2\}$, $(\ell_0^k, \dots, \ell_j^k)$ must be a leximin² allocation for $M_{\ell_j^k}$ with j agents. Thus if $\ell_j^1 = \ell_j^2$ for some $j \in N$, then $\ell_k^1 = \ell_k^2$ for all $k \leq j$.

Suppose, for the sake of contradiction, that there exists $j \in N$ such that $\ell_j^1 > \ell_j^2$. Let k be the first index larger than j such that $\ell_k^1 < \ell_k^2$. Since $\ell_n^1 = t < t+1 = \ell_n^2$, such k must exist and, in the same time, satisfies that $\ell_{k-1}^1 > \ell_{k-1}^2$. For $p, q \in \{1, 2\}$, define A^{pq} as the allocation for $M_{\ell_k^q}$ such that $P(A^{pq}) = (\ell_0^p, \ell_1^p, \dots, \ell_{k-1}^p, \ell_k^q)$. Since both A^{11} and A^{22} are leximin², we have

$$Q(A^{21}) \preceq Q(A^{11}),$$

and

$$Q(A^{12}) \preceq Q(A^{22}). \quad (10)$$

It suffices to show that $Q(A^{11}) \preceq Q(A^{21})$ and $Q(A^{22}) \preceq Q(A^{12})$. Since in this case, if

$$(\ell_0^2, \ell_1^2, \dots, \ell_{k-1}^2) \preceq (\ell_0^1, \ell_1^1, \dots, \ell_{k-1}^1),$$

then we can replace the first k elements of $P(X^t) = (\ell_0^1, \dots, \ell_n^1)$ with $\ell_0^2, \dots, \ell_{k-1}^2$ to obtain another leximin allocation A with $P(A) \preceq P(X^t)$, which contradicts the fact that X^t is leximin². By contrast, if

$$(\ell_0^1, \ell_1^1, \dots, \ell_{k-1}^1) \preceq (\ell_0^2, \ell_1^2, \dots, \ell_{k-1}^2),$$

then we can replace the first k elements of $P(X^{t+1}) = (\ell_0^2, \dots, \ell_n^2)$ with $\ell_0^1, \dots, \ell_{k-1}^1$ to obtain another leximin allocation A with $P(A) \preceq P(X^{t+1})$, which contradicts the fact that X^{t+1} is leximin².

Due to the symmetry, we only prove $Q(A^{11}) \preceq Q(A^{21})$. Since $\ell_k^1 < \ell_k^2$ and $\ell_{k-1}^1 > \ell_{k-1}^2$, we have

$$v(A_k^{11}) < v(A_k^{12}) < v(A_k^{22}), \quad (11)$$

and thus

$$v(A_k^{11}) = v(A_k^{12}) - v(M_{\ell_k^2} \setminus M_{\ell_k^1}) < v(A_k^{22}) - v(M_{\ell_k^2} \setminus M_{\ell_k^1}) = v(A_k^{21}). \quad (12)$$

Note that all but the last blocks in A^{12} are the same as those in A^{11} , and all but the last blocks in A^{22} are the same as those in A^{21} . Recall that when deciding the lexicographic order of two tuples, we first compare the first elements of two tuples; if they are equal, then compare the second ones, and so on. If (10) is decided only by the blocks in A^{12} and A^{22} with valuations smaller than $v(A_k^{11})$, then $Q(A^{11}) \preceq Q(A^{21})$ holds because the blocks with valuations smaller than $v(A_k^{11})$ in A^{11} and A^{21} are the same with those in A^{12} and A^{22} . On the other hand, if (10) is decided by the blocks with valuations at least $v(A_k^{11})$, then (10) implies that the number of blocks with valuations $v(A_k^{11})$ in A^{12} is not smaller than the number of blocks with valuations $v(A_k^{11})$ in A^{22} . By (11) and (12), the number of blocks with valuations $v(A_k^{11})$ in A^{11} is strictly larger than the number of blocks with valuations $v(A_k^{11})$ in A^{21} . Therefore, $Q(A^{11}) \preceq Q(A^{21})$ is established.

All the above concludes the proof of Lemma 4.7.

D Proof of Theorem 5.2

Proof. Suppose that there are T items and n agents with valuation functions v_1, \dots, v_n . For any $t \in [T]$, let

$$v_i(g_t) = \begin{cases} n^{2c}, & 3cn + 3(i-1) < t \leq 3cn + 3i \text{ for some } c \geq 0, \\ 0, & \text{otherwise,} \end{cases}$$

for all $i \in N$. That is, each period of length $3n$ is divided into n blocks of length 3, and agent i is only interested in the items in the i -th block in each period c with n^{2c} valuation for each of these items. It suffices to show that, for every $k \geq 2n$, the first $3k - 3n$ items belong to one agent in round $3k$ and belong to a different agent in round $3k + 3$. Since in this case, if we assume that $T \gg n$, the number of adjustments required is at least $\sum_{k=2n}^{T/3-1} 3(k-n) = \Theta(T^2)$.

We only give the proof for k such that k is a multiple of n , and the proof can be easily generalized to any $k \geq 2n$. Now we prove that, for any $c \geq 2$ and $k = cn$, the first $3k - 3n = (c-1) \cdot 3n$ items belong to agent 1 in round $3k$ and belong to agent 2 in round $3k + 3$. Define $G_i = \{g_t \mid (c-1) \cdot 3n + 3(i-1) < t \leq (c-1) \cdot 3n + 3i\}$ as the set of items that arrive during period c and agent i is interested in. In round $3k$, if agent i gets none of the items in G_i , there must exist another agent j that obtains at least two of them due to the contiguity requirement. Furthermore, the total valuation of the first $3k - 3n = (c-1) \cdot 3n$ items for agent i is

$$3 \sum_{j=0}^{c-2} n^{2j} = 3 \cdot \frac{n^{2(c-1)} - 1}{n^2 - 1} < n^{2(c-1)} = v_i(g), \quad \forall g \in G_i.$$

Thus if agent i gets none of the items in G_i , he must envy agent j up to one item. As a result, to satisfy EF1, each agent i should get at least one of the items in G_i . Due to the contiguity requirement, agent i must get the i -th block which contains at least one item in G_i . In this case, the first $3k - 3n = (c-1) \cdot 3n$ items belong to agent 1. Similarly, we can show that in round $3k + 3$, the first $3k - 3n + 3$ items belong to agent 2 and we are done.

Therefore, we conclude that any EF1 algorithm requires at least $\Omega(T^2)$ adjustments. \square