

Non-Monotonicity in Fair Division of Graphs

Hadi Hosseini
Penn State University
hadi@psu.edu

Shraddha Pathak
Penn State University
ssp5547@psu.edu

Yu Zhou*
Beijing Normal University
yu.zhou@bnu.edu.cn

Abstract

We consider the problem of fairly allocating the vertices of a graph among n agents, where the value of a bundle is determined by its *cut value*—the number of edges with exactly one endpoint in the bundle. This model naturally captures applications such as team formation and network partitioning, where valuations are inherently *non-monotonic*: the marginal values may be positive, negative, or zero depending on the composition of the bundle. We focus on the fairness notion of envy-freeness up to one item (EF1) and explore its compatibility with several efficiency concepts such as Transfer Stability (TS) that prohibits single-item transfers that benefit one agent without making the other worse-off. For general graphs, our results uncover a non-monotonic relationship between the number of agents n and the existence of allocations satisfying EF1 and transfer stability (TS): such allocations always exist for $n = 2$, may fail to exist for $n = 3$, but exist again for all $n \geq 4$. We further show that existence can be guaranteed for any n by slightly weakening the efficiency requirement or by restricting the graph to forests. All of our positive results are achieved via efficient algorithms.

1 Introduction

Fair division of indivisible items is a fundamental problem in multiagent systems and computational social choice. A central goal in fair division is to allocate a set of indivisible, non-shareable items among (potentially heterogeneous) agents in a manner that is both fair and efficient, guided by well-established axiomatic principles. However, much of the existing literature assumes that agents have additive, or otherwise *monotonic* valuations. These assumptions often fail to capture more realistic scenarios where the value of an item is non-monotonic and may depend on the composition of the bundle it belongs to.

In this paper, we initiate a formal study of non-monotonic *bundle-dependent* valuations, where the marginal value of an item may vary—potentially becoming negative—depending on the current bundle held by the agent. For instance, in settings such as team formation or committee selection with diverse expertise, the value of an additional member, such as a goalkeeper, may vary significantly depending on the existing composition: they may be essential for a team lacking a substitute, but redundant or even burdensome for a fully staffed team due to added financial cost.

We focus on a specific class of bundle-dependent preferences based on *cut-valuations* defined over undirected graphs. In this model, items correspond to vertices of a graph $G = (V, E)$, and an agent’s value for a bundle is determined by the number of edges that are cut, i.e. edges with exactly one endpoint in the bundle. Thus, the marginal value of an item is positive for bundle

*Corresponding author. This work was done when Yu Zhou was a Postdoc at Penn State University.

S if it has less neighbors inside S than outside S , otherwise it is zero or negative. This yields valuations that are inherently *non-monotonic*: adding an item can increase or decrease an agent’s utility depending on the underlying structure of the bundle.

We are interested in allocating vertices of the graph (corresponding to items) in a fair way while guaranteeing some notion of economic efficiency among the agents. Cut-valuations naturally model the aforementioned team formation problems that value fairness and diversity, where vertices represent individuals in a social or similarity graph and edges capture social ties or overlapping expertise. These non-monotonic valuations can also be seen as a soft relaxation of conflict constraints commonly studied in fair division [Hummel and Hetland, 2022, Chiarelli et al., 2020, Li et al., 2021]. Such interplay between constraint-based models and their valuation-based counterparts have been previously considered for matroids [Biswas and Barman, 2019, Cookson et al., 2025, Barman and Verma, 2020], as well as cardinality constraints and level valuations [Biswas and Barman, 2018, Christodoulou and Christoforidis, 2024].

1.1 Our Contributions

We consider the problem of partitioning general graphs, as well as forests, with a focus on simultaneously achieving fairness and economic efficiency. For fairness, we consider *envy-freeness* (EF), which informally requires that no agent strictly prefers another agents’ bundle; we also consider its well-studied relaxation, *envy-freeness up to one* (EF1), which requires that any pairwise envy is eliminated by removal of a single item. Under cut-valuations, allocating all vertices to a single agent is vacuously envy-free: all agents receive a value of zero. To avoid such degenerate outcomes, we study fairness in conjunction with efficiency, including: (i) *social optimality* (SO), which maximizes total utility; (ii) *Pareto optimality* (PO), where no agent can be made better off without harming another. We also propose two natural and desirable efficiency notions: (iii) *transfer stability* (TS), prohibiting single-item transfers that benefit one agent without making the other worse-off; and (iv) *weak transfer stability* (WTS), forbidding transfers that strictly benefit both agents involved.¹

General graphs. Our first set of results reveal an axiomatic non-monotonic relationship between existence guarantees for EF1+TS allocation and the number of agents n under cut-valuations: They exist for $n = 2$ (Proposition 1), may fail to exist for $n = 3$ (Theorem 1), but exist again for all $n \geq 4$ (Theorem 2). The $n = 3$ counter-example also shows the incompatibility of EF1 with PO and SO. In contrast, the existence of EF1 and SO allocations follows a monotonic pattern: They always exist for $n = 2$, but may not exist when $n \geq 3$ (Theorem 5).

This raises a natural question: *Is there an efficiency notion that is always compatible with EF1, regardless of n ?* We show that a slight relaxation of TS (WTS) is the strongest efficiency notion which can always be guaranteed alongside EF1, irrespective of n (Theorem 3). Moreover, such allocations can be computed in polynomial time. We also show that allocations that are $\frac{1}{2}$ -EF1, a multiplicative approximation of EF1, can co-exist alongside PO for arbitrary n (Proposition 2).

We note that our EF1+WTS result (Theorem 3) also improves upon prior guarantees for equitable graph cut. Specifically, we show that the vertices of a graph can be partitioned into n parts such that the absolute difference between the cut-values of any two parts is bounded by the maximum degree Δ in the graph (Corollary 1). This improves on the previous bound of $5\Delta + 1$ by Barman and Verma [2025].

¹In Section A, we demonstrate that techniques developed for monotone valuations fail to extend to the cut-valuation setting.

Forests. When the underlying graph is a forest (i.e. acyclic), we show that EF1 and SO allocations always exist and can be computed in polynomial time (Theorem 4). Interestingly, the standard techniques from the monotone settings—iteratively building toward an EF1 allocation where, on assigning each new item, EF1 is maintained as an invariant—do not apply to such special cases either. While our algorithm also iteratively builds towards an EF1 allocation, there are situations where we temporarily break intermediate EF1 guarantees, but ensure that such violations can be ‘caught-up’ through subsequent assignments.

Finally, we note that the negative counter-example we construct for showing the incompatibility of EF1 alongside various fairness notions under general graphs is in fact a *complete bipartite graph* with treewidth 2. This highlights that while strong existence and algorithmic guarantees hold for forests (bipartite graphs; treewidth 1), they break even for graphs that are only slightly different.

1.2 Related works

Fair Division under Non-monotone Valuations Early works on non-monotone valuations focus on *doubly monotone* (often *additive*) valuations, where each item is either always a good (positive value) or a chore (negative value) for an agent, regardless of the bundle. In such settings, EF1 allocations always exist [Aziz et al., 2022, Bhaskar et al., 2020]. The existence of EF1+PO allocations remains open, even in the simple additive case [Liu et al., 2024]. While such allocations are known for (additive) goods-only instances [Caragiannis et al., 2019], their existence in chores-only settings was only recently established [Mahara, 2025]. Recent work either relaxes EF1 [Barman et al., 2025] or focuses on specific valuations [Hosseini et al., 2023]. More general non-monotone settings—where items act as goods or chores depending on the bundle—have been recently explored under special classes such as subadditive, non-negative valuations², identical trilean, and separable single-peaked [Barman and Verma, 2025, Bilò et al., 2025, Bhaskar et al., 2025].

Fair Division of Graphs Graphs appear in fair division both as feasibility constraints (e.g. conflicting or connectivity [Hummel and Hetland, 2022, Chiarelli et al., 2020, Li et al., 2021, Kumar et al., 2024, Igarashi et al., 2025, Bouveret et al., 2017, Bei et al., 2022, Bilò et al., 2022]), and to define agents’ valuations (e.g. from graphical structures such as shortest paths [Hosseini et al., 2025, Hosseini and Schierreich, 2025], minimum vertex cover [Wang and Li, 2024], or edge-based models where each item (an edge) is valued by exactly two agents (the incident vertices) [Christodoulou et al., 2023, Deligkas et al., 2024, Zhou et al., 2024]). A related line of research considers partitioning of friends into groups that are balanced and fair [Li et al., 2023, Deligkas et al., 2025]. Here, an agents utilities are monotone and based on intra-group connections. In contrast, our model captures non-monotonic, bundle-dependent preferences, arising in problems where inter-group connections are valued.

2 Preliminaries

Given any $j, k \in \mathbb{N}$ such that $j \leq k$, let $[k] := \{1, \dots, k\}$, and $[j, k] := \{j, j+1, \dots, k\}$.

An instance of the cut-valuation problem is denoted by $\langle N, G, v \rangle$, where $N = \{1, 2, \dots, n\}$ is the set of n agents, $G = (V, E)$ is an undirected graph with $|V| = m$ vertices and $|E|$ edges, and a cut-valuation v . The vertex set V corresponds to the set of items to be distributed among the

²We note that cut-valuations studied in this paper are a subclass of non-negative and were discussed in [Barman and Verma, 2025].

agents; throughout the paper, we use items and vertices interchangeably. The edge between any two adjacent vertices o, o' is denoted by $e = (o, o')$. The neighbors of o within the set S is denoted by $N_S(o) := \{o' \in S : (o, o') \in E\}$. The degree of o , denoted by $\deg(o) := |N_V(o)|$, is the number of its neighbors in G . A graph is a forest if it is acyclic. Note that a graph may contain multiple disconnected components.

Cut-Valuations. The cut-valuation $v : 2^V \rightarrow \mathbb{R}$ defines the cardinal preferences of the agents for any subset of items in V . Specifically, the cut-value of any subset S is $v(S) := |\{(o, o') \in E : o \in S \text{ and } o' \in V \setminus S\}|$, which is the number of edges that have one endpoint in S and the other endpoint in $V \setminus S$. Crucially, the cut-valuation represents a subclass of *non-monotone* valuations, where an item o has a *positive marginal value* for a bundle S if and only if o has more neighbors outside of S than in S , i.e. $|N_S(o)| < |N_{V \setminus S}(o)|$. Otherwise, o may have a *negative or non-positive marginal value*. For the ease of exposition, we refer to an item with positive marginal value (with respect to a subset) as a *good*, otherwise it is called a *chore* (with respect to the subset). Without loss of generality, we assume (i) $|V| > n$, and (ii) G has no singleton components because such vertices have zero value and can be assigned arbitrarily.

Allocation. An allocation $A = (A_1, \dots, A_n)$ is a partition of vertices (items) into n pairwise disjoint subsets of V , i.e., for every $A_i, A_j \subseteq V$, $A_i \cap A_j = \emptyset$. Each part A_i contains a bundle of items assigned to agent $i \in N$. Note that the items in A_i may not be adjacent. An allocation may be *partial* when $\cup_{i \in N} A_i \subsetneq V$, or *complete* when $\cup_{i \in N} A_i = V$.

Let us review the cut-valuations and potential allocations using Example 1.

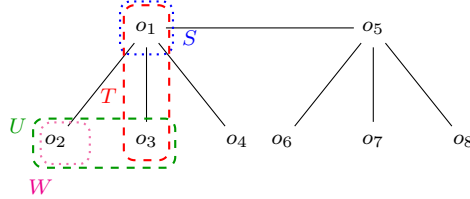


Figure 1: Graph with 8 items (vertices).

Example 1 (Allocations and cut-valuations). Consider the graph in Figure 1. The cut-value of the set $S = \{o_1\}$ is $v(S) = 4$. However, the set T that is obtained by adding an item o_3 to S reduces the cut-value to $v(T) = 3$. In other words, o_3 is a chore for the set S . However, o_3 is a good for set $W = \{o_2\}$ since addition of o_3 to W results in the set U that has a cut-value of $v(U) = 2$, whereas $v(W) = 1$.

Remark 1 (Relation to Graph Partitions). In graph theory, a k -partition of a graph $G = (V, E)$ is a division of V into k disjoint subsets $V = V_1 \cup \dots \cup V_k$, where $V_i \cap V_j = \emptyset$ for all $i \neq j$, where V_i may be subject to additional structural constraints such as being connected subgraphs or forming independent sets. In our model, an allocation corresponds to a complete partition of V into n bundles $(A_1, \dots, A_n)$, one for each agent. Our framework does not impose a connectivity constraint, i.e. multiple disconnected components may be allocated as a bundle (say A_i) to an agent.

Fairness. An allocation $A = (A_1, \dots, A_n)$ is **envy-free** (EF) if every agent values their own bundle at least as much as that of any other agent's bundle, i.e., $v(A_i) \geq v(A_j)$ for every two agents $i, j \in N$. An allocation A is **envy-free up to one item** (EF1) if for every pair of agents $i, j \in N$

where $v(A_j) > v(A_i)$, there exists an item $o_j \in A_j$ such that $v(A_i) \geq v(A_j \setminus \{o_j\})$ ³ [Lipton et al., 2004, Budish, 2011].

In Example 1, although $v(T) > v(W)$, the (hypothetical) removal of $o_1 \in T$ eliminates the envy. Note that in this setting, the existence of an EF1 allocation is particularly delicate due to the non-monotonicity of valuations since removing a single item (vertex) may sometimes increase the agent’s value. In this example, the removal of $o_3 \in T$ in fact increases the value of T from 3 to 4.

Efficiency. In our setting, fairness alone may not be sufficient to rule out undesirable allocations. For instance, allocating all vertices to a single agent is envy-free, however, all agents receive a value of zero. Thus, we consider *non-empty* allocations wherein each agent receives at least one item and define several plausible efficiency notions.

An allocation, A , is **Social Optimal** (SO) if it maximizes the utilitarian social welfare, i.e., $A \in \arg \max_{X \in \Pi_n(V)} \sum_{i \in N} v(X_i)$, where $\Pi_n(V)$ denotes the set of all n -partitions of V . An allocation A is **Pareto Optimal** (PO) if it is not Pareto dominated by another allocation, that is, no other allocation B exists such that $v(B_i) \geq v(A_i)$ for every $i \in N$ with at least one inequality being strict.

An allocation, $A = (A_1, \dots, A_n)$, is **Transfer Stable** (TS) if, for every $i, j \in N$, there does not exist an item $o \in A_i$ such that $v(A_i \setminus \{o\}) \geq v(A_i)$ and $v(A_j \cup \{o\}) \geq v(A_j)$ with at least one inequality being strict. Allocation A is **Weak Transfer Stable** (WTS) if, for every $i, j \in N$, there does not exist an item $o \in A_i$ such that $v(A_i \setminus \{o\}) > v(A_i)$ and $v(A_j \cup \{o\}) > v(A_j)$. Informally, a TS allocation requires that no single item can be transferred from an agent to another without making either agent worse-off while increasing the value of at least one of the two agents. It is WTS if no such transfer exists that necessarily makes both agents strictly better-off. It is easy to see that $\text{SO} \implies \text{PO} \implies \text{TS} \implies \text{WTS}$ while the converse directions do not hold; see Example 2.

Example 2 (Understanding Efficiency Notions). *Let us revisit the example given in Figure 1 and consider 7 agents. The allocation $A = (\{o_1, o_6\}, \{o_2\}, \{o_3\}, \{o_4\}, \{o_5\}, \{o_7\}, \{o_8\})$ is SO. The allocation $(\{o_1, o_5\}, \{o_2\}, \{o_3\}, \{o_4\}, \{o_6\}, \{o_7\}, \{o_8\})$ is PO but not SO. For an example of an allocation that is TS but not PO, consider the same graph but let there be $n = 4$ agents. The allocation $(\{o_1, o_5\}, \{o_2, o_6\}, \{o_3, o_7\}, \{o_4, o_8\})$ satisfies TS but is not PO since it is Pareto dominated by the allocation $(\{o_1, o_6, o_7\}, \{o_5\}, \{o_2, o_3\}, \{o_4, o_8\})$. Finally, to illustrate an allocation that is WTS but not TS, consider a cycle with 6 nodes $(o_1, o_2, \dots, o_6)$ and $n = 3$ agents. The allocation in which two consecutive nodes (say $(\{o_1, o_2\}, \{o_3, o_4\}, \{o_5, o_6\})$) are assigned to the same agent is WTS but not TS.*

3 General Graphs

In this section, we study general graphs that may contain cycles. Our first set of results reveal a non-monotonic relationship between existence guarantees for EF1+TS allocation and the number of agents under cut-valuations: They exist for $n = 2$, may fail to exist for $n = 3$, but exist again for all $n \geq 4$. Interestingly, this non-monotonic dependence is not present for SO. In Section B, we provide a family of instances for every $n \geq 3$ agents such that no allocation is simultaneously EF1 and SO.

3.1 Two Agents: Existence

To warm up, we start by considering the case of $n = 2$, and give an existence guarantee for EF and SO: Since the cut function of a graph G is symmetric, i.e. $v(S) = v(V \setminus S)$ for any $S \subseteq V$, any

³We refer the reader to Section A for a discussion on alternative definitions of EF1 used in the literature.

partition between two agents is EF. Moreover, an allocation is SO if and only if it is PO, because the cut-values are symmetric and an increase in the value of one agent leads to an increase in value for both.

Proposition 1. *When $n = 2$, an allocation satisfying EF and SO always exists, but computing such an allocation is NP-HARD. Moreover, an allocation satisfying EF and TS always exists and can be computed in polynomial time.*

Proof. For two agents, every allocation is EF due to symmetry of cut-valuations. Thus, the problem reduces to finding an SO partition. The symmetry of valuations means that the partition corresponding to the maximum cut is SO. That is, a MAX-CUT results in EF+SO; however, since computing a MAX-CUT is NP-HARD [Garey et al., 1974], the hardness follows.

Focusing on TS, we devise a simple greedy algorithm as follows: Start with an arbitrary 2-partition of the graph, say $A = (V, \emptyset)$, and iteratively transfer an item from one bundle to the other such that the value of the bundles increases. The value can increase at most $O(m^2)$ times since, the value increases by at least 1 in each round and for any set S , $v(S) \leq |E|$ where $|E| \in O(m^2)$ for simple graphs. Again, by symmetry of cut-valuations for 2-partitions, the final allocation after the greedy process is also EF. $\square$

3.2 Three Agents: Non-existence

While for $n = 2$, EF+TS always admits a polynomial-time solution, we show that for $n = 3$, even EF1—a strictly weaker notion than EF—is incompatible with TS (and thus, with PO and SO as well).

Theorem 1 (Non-existence of EF1+TS allocations). *There exists a cut-valuation instance with $n = 3$ agents where no EF1 allocation is TS (and thus PO and SO).*

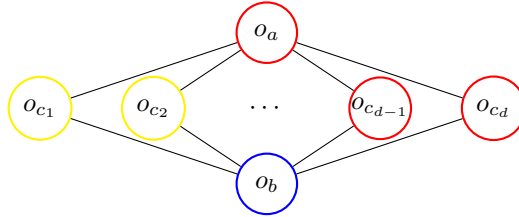


Figure 2: An example showing the incompatibility of EF1 and transfer stability (TS) for $n = 3$.

Proof. Consider the instance with 3 agents and $d + 2$ items, o_a, o_b and o_{c_t} where $t \in [d]$ as shown in Figure 2. Suppose $d \geq 3$ and that d is odd. In any EF1 allocation, o_a and o_b must be allocated to different agents; if not, i.e., if $A_1 = \{o_a, o_b\}$, then one of the remaining two agents will receive a value of at most $2\lfloor \frac{d}{2} \rfloor < d = v(A_1 \setminus \{o_a\}) = v(A_1 \setminus \{o_b\})$, which violates EF1. Thus, o_a is allocated to agent 1 (red) and o_b to agent 2 (blue).

After allocating $\lceil \frac{d}{2} \rceil$ of the remaining items to the third agent (yellow), no new item can be allocated to A_3 without breaking EF1 because each item has marginal value of two for agent 3, and on addition of any new item $o \in \{o_{\lceil \frac{d}{2} \rceil + 1}, \dots, o_{cd}\}$, we have $v(A_3 \cup \{o\} \setminus \{o'\}) = 2\lceil \frac{d}{2} \rceil > d = v(A_1) = v(A_2)$ for any $o' \in A_3 \cup \{o\}$. Thus, all the remaining items must be (wastefully) allocated to either agent 1 or 2 in order to maintain EF1. In Figure 2, they are arbitrarily allocated to agent 1 (red). But the marginal values of these items are 0 for both agents 1 and 2; thus, to maintain TS these items should be transferred to agent 3, which whereas violates EF1. $\square$

While Theorem 1 shows the incompatibility of EF1 with TS, PO, and SO, the EF1 allocation marked in Figure 2 satisfies the weaker WTS condition. In fact, as we show in Section 3.4, allocations that are both EF1 and WTS always exist and can be computed in polynomial time.

3.3 Four or More Agents: Existence

Surprisingly, when $n \geq 4$, EF1 and TS allocations always exist and can be computed in polynomial time.

Theorem 2 (EF1+TS allocations when $n \geq 4$). *When $n \geq 4$, a complete allocation satisfying EF1 and TS always exists and can be computed in polynomial time.*

Before presenting the main algorithm for our constructive proof, we observe that, under identical valuations, if the allocation is ordered from the least to the highest valued bundle, then the allocation is EF1 *if and only if* the agent with the least-valued bundle is not involved in any EF1 violation; we refer the readers to Section C for the proof. This lemma enables us to restrict attention to EF1 violations involving only the agent with the least-valued bundle.

Lemma 1. *Given an instance with identical valuations and allocation $A = (A_1, \dots, A_n)$ such that $v(A_1) \leq \dots \leq v(A_n)$, A is EF1 if and only if, for every $j \in N$ such that $v(A_j) > v(A_1)$, there exists an item $o_j \in A_j$ s.t. $v(A_1) \geq v(A_j \setminus \{o_j\})$.*

Algorithm 1 operates over the space of complete allocations and performs local search to eliminate EF1 violations. A subroutine (Algorithm 2) is repeatedly invoked to enforce TS by reassigning items whose marginal value to the current holder is weakly negative.

Algorithm 1 Computing EF1 + TS allocations on general graphs when $n \geq 4$

- 1: Initialize $A = (A_1, \dots, A_n)$ arbitrarily with any complete allocation.
- 2: Relabel bundles s.t. $v(A_1) \leq \dots \leq v(A_n)$.
- 3: $A \leftarrow \text{TS-SUBROUTINE}(\text{None}; A)$
- 4: **while** ($\exists$ EF1 violation from agent 1 to some other agent) **do**

▷ Case I (Lines 5–9): When there exists an item in an EF1-violator’s bundle with positive marginal value for A_1 .

- 5: **while** (there exists EF1 violation towards an agent i , s.t. $v(A_1 \cup \{o\}) > v(A_i)$ for some $o \in A_i$) **do**
- 6: Transfer item o to A_1
- 7: Relabel bundles s.t. $v(A_1) \leq \dots \leq v(A_n)$
- 8: $A \leftarrow \text{TS-SUBROUTINE}(\text{None}; A)$
- 9: **end while**

▷ Case II (Lines 10–14): When all items in the EF1-violator’s bundle have non-positive marginal value for A_1 .

- 10: **while** (there exists EF1 violation from agent 1 to special agent i^* and $v(A_1) \geq v(A_1 \cup \{o\})$ for every $o \in A_{i^*}$) **do**
 - 11: Transfer any such item o to another agent, $k \neq i^*$, s.t. $v(A_k \cup \{o\}) > v(A_k)$
 - 12: **end while**
 - 13: Relabel bundles s.t. $v(A_1) \leq \dots \leq v(A_n)$.
 - 14: $A \leftarrow \text{TS-SUBROUTINE}(i^*; A)$
 - 15: **end while**
-

Algorithm Description. Algorithm 1 starts with any arbitrary complete allocation. Since agents have identical valuation functions, any bundle can be assigned to any agent. Thus, throughout the algorithm, the bundles are sorted in non-decreasing order of value i.e. $v(A_1) \leq v(A_2) \leq \dots \leq v(A_n)$. The algorithm proceeds by iteratively resolving pairwise EF1 violations in two cases: In **Case I**, we run a *local search* repeatedly to find a bundle, say A_i , such that i) A_i violates EF1 with respect to the lowest valued bundle A_1 , and ii) there exists an item $o \in A_i$ with strictly positive marginal value for agent 1, i.e. $v(A_1 \cup \{o\}) > v(A_1)$. In this case, we transfer the item to A_1 , update the labels, and repeat. Otherwise, we are in the case (**Case II**) where agent 1's violation of EF1 towards another bundle, say A_{i^*} , cannot be remedied by transferring an item from A_{i^*} to A_1 since all items in A_{i^*} are of a non-positive marginal value for agent 1. In this case, the algorithm transfers an item $o' \in A_{i^*}$ to another bundle, say A_k , for whom $v(A_k \cup \{o'\}) > v(A_k)$. This process repeats until there is no EF1 violations from agent 1 to i^* . The key observation is that when the algorithm is in Case II, there is an EF1 violation with respect to only a single agent (see Claim 2 below); thus, it suffices to consider only a single special agent i^* . Note that to maintain efficiency, after each transfer, the algorithm invokes the TS subroutine (Algorithm 2) described next.

Maintaining Efficiency TS-Subroutine. This subroutine (Algorithm 2) takes as input a complete allocation and a designated *special agent* (call it i^*), and returns a TS allocation in which no agent is worse-off compared to the input allocation. The special agent is used solely to ensure that, in certain cases, its bundle remains unchanged; which becomes critical when arguing the convergence of the algorithm (see the next paragraph). The subroutine iteratively identifies an agent holding an item with non-positive marginal value and transfers that item as follows: If the agent with the non-positive marginal value item is the one with the least-valued bundle (aka agent 1), the item is transferred to any bundle (excluding the special agent i^*) where it has strictly positive marginal value; otherwise, it is transferred to the least-valued bundle i.e., A_1 , if it provides strictly positive marginal value there; if not, it is transferred to any agent (excluding the special agent i^*) for whom the item has strictly positive marginal value. This process continues until no such item remains, yielding a TS allocation.

Algorithm 2 Transfer Stability (TS-SUBROUTINE)

Input: Special agent i^* and a complete allocation $A = (A_1, \dots, A_n)$ s.t. $v(A_1) \leq \dots \leq v(A_n)$

Output: A TS allocation s.t. $v(A_1) \leq \dots \leq v(A_n)$

- 1: **while** (there exists an agent $i \in N$ with an item $o \in A_i$ s.t. $v(A_i) \leq v(A_i \setminus \{o\})$) **do**
 - 2: If $i = 1$ (least valued bundle), then transfer o to an agent $k \neq i^*$ s.t. $v(A_k \cup \{o\}) > v(A_k)$;
 - 3: Else if $i \neq 1$ and $v(A_1 \cup \{o\}) > v(A_1)$, then transfer o to A_1 ;
 - 4: Else transfer o to any agent $k \neq i^*$ s.t. $v(A_k \cup \{o\}) > v(A_k)$.
 - 5: Relabel the bundles s.t. $v(A_1) \leq \dots \leq v(A_n)$.
 - 6: **end while**
-

The Importance of $n \geq 4$ and Special Agent. The structure of cut-valuations imposes a limit on the number of agents who can view a given item as a chore (non-positive marginal value); as we show in Lemma 2, this can be at most two such agents. This means that when $n \geq 3$, in a TS allocation, each item must be assigned to an agent who values it positively. As shown in Theorem 1, sometimes this constraint conflicts with the EF1 requirement.

When $n \geq 4$, this bound on the number of agents guarantees the existence of *multiple* agents for whom an item is positively valued. This enables series of transfer to obtain EF1 when direct transfers to the least valued agent may not be possible: An item can be transferred to another

agent who values it positively, and if needed, another item from their bundle can be passed to A_1 . To ensure polynomial-time termination, we sometimes need to treat an agent as “special” and not alter its bundle during the TS-SUBROUTINE. In such cases, we again rely on the $n \geq 4$ condition to guarantee that there exists another agent—distinct from the special agent—who values an item positively.

Lemma 2. *Given a cut-valuation instance and a (partial) allocation, any item $o \in V$ has non-positive marginal value for at most two agents. Moreover, when $n \geq 4$, item o has positive marginal value for at least two agents.*

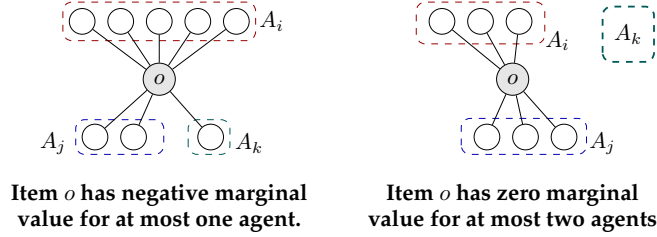


Figure 3: An illustration of the characterization in Lemma 2.

Proof. Suppose o has a negative marginal value for agent i . This means that o has more neighbors inside A_i than outside A_i , i.e., $|N_{A_i}(o)| > |N_{V \setminus A_i}(o)|$. For any other agent $j \neq i$, this implies that $N_{A_j}(o) \subseteq N_{V \setminus A_i}(o)$ and $N_{A_i}(o) \subseteq N_{V \setminus A_j}(o)$. Thus, $|N_{A_j}(o)| \leq |N_{V \setminus A_i}(o)| < |N_{A_i}(o)| \leq |N_{V \setminus A_j}(o)|$, i.e., o has fewer neighbors inside A_j than outside, so o must have a positive marginal value (i.e. be a good) for all such j . For an illustration of this, see Figure 3.

Similarly, if agent i has zero marginal value for some $o \in V$ (that is, $|N_{A_i}(o)| = |N_{V \setminus A_i}(o)|$ i.e. the number of neighbors of o inside and outside A_i is equal), then at most one other agent (say j) can have zero value for o (when $N_{A_j}(o) = N_{V \setminus A_i}(o)$; see Figure 3), and all other agents must have strictly a positive marginal value for o . This proves the first part of the lemma, i.e. any item has a non-positive marginal value for at most two agents. By the pigeonhole principle, when $n \geq 4$, then there are at least two agents for whom o has a positive marginal value. $\square$

Before proving Theorem 2, we first present three key technical claims. First, we show that each agent derives at least half the total value of all bundles that contain only items with non-positive marginal value to that agent. Intuitively, consider one such chore o that is received by an agent j . Since o is a chore for i , there are at least as many edges from o to A_i as from o to $V \setminus A_i$. These edges (from o to A_i) contribute to the value of A_i . On the other hand, the value of A_j can be at most equal to sum of all edges from $o \in A_j$, which is at most twice the number of edges from o to A_i .

Claim 1. *For an agent $i \in N$, consider any set of bundles X such that for every bundle $A_k \in X$, it holds that $v(A_i) \geq v(A_i \cup \{o\})$ for every $o \in A_k$. Then, we have*

$$v(A_i) \geq \frac{1}{2} \sum_{A_k \in X} v(A_k).$$

Proof. First, we note that when $o \in A_k$ is a chore with respect to A_i , we have

$$|\{(o, o') \in E : o' \in A_i\}| \geq |\{(o, o') \in E : o' \in V \setminus A_i\}|$$

Adding $|\{(o, o') \in E : o' \in A_i\}|$ on both sides, we get

$$2|\{(o, o') \in E : o' \in A_i\}| \geq |\{(o, o') \in E : o' \in A_i\}| + |\{(o, o') \in E : o' \in V \setminus A_i\}|$$

Dividing both sides by 2, we have

$$|\{(o, o') \in E : o' \in A_i\}| \geq \frac{1}{2}|\{(o, o') \in E : o' \in V\}| \quad (1)$$

Now, using (1), we can lower bound the value of A_i in the following manner.

$$\begin{aligned} v(A_i) &= |\{(o, o') \in E : o \in A_i, o' \in V \setminus A_i\}| \\ &\geq \sum_{A_k \in X} |\{(o, o') \in E : o \in A_i, o' \in A_k\}| && \text{(Since } A_k \subseteq V \setminus A_i) \\ &\geq \frac{1}{2} \sum_{A_k \in X} \sum_{o \in A_k} |\{(o, o') \in E : o' \in V\}| && \text{(From (1))} \\ &\geq \frac{1}{2} \sum_{A_k \in X} v(A_k), && \text{(Since } v(S) \leq \sum_{o \in S} \deg(o)) \end{aligned}$$

thus proving the claim. $\square$

Using this claim, it is easy to see that when Case II happens, i.e., when there is EF1 violation from agent 1 to some agent i^* but A_{i^*} does not have any item with positive marginal value for A_1 , such a violation can only be towards a *single* agent. Thus, all remaining bundles contain at least one item with positive marginal value.

Claim 2. *Let X be the set of all bundles towards which the first agent (agent with least valued bundle) has EF1 violations. If, for every bundle $A_k \in X$ and $o \in A_k$ we have $v(A_1) \geq v(A_1 \cup \{o\})$, then $|X| \leq 1$. Moreover, for any other bundle $A_j \notin X$, there exists an item $o' \in A_j$ such that $v(A_1 \cup \{o'\}) > v(A_1)$.⁴*

Proof. Let i^* be an agent who is EF1-envied by agent 1. From Case 2, we know that every item in A_{i^*} must be a (weak or strict) chore with respect to A_1 . If there exists another agent (say k) whose items are all chores for agent 1, by Claim 1, we have that

$$v(A_1) \geq \frac{1}{2} \cdot (v(A_j) + v(A_k)),$$

which is a contradiction since agent 1 strictly envies i^* and $v(A_1) \leq v(A_k)$. Therefore, in Case 2 where all items in A_{i^*} are chores with respect to A_1 , it must be the case that i^* is the only agent who is EF1-envied by agent 1. $\square$

To argue about the termination of the main algorithm, Algorithm 1, we use a lexicographic improvement in the vector $\Phi = (v(A_1), -n_1)$, where $v(A_1)$ represents the value of the least-valued bundle and n_1 represents the number of agents with this least-value. Finally, to prove that the TS-SUBROUTINE (Algorithm 2) terminates in polynomial time, we again use a potential argument; in particular, we show that the social welfare strictly increases and Φ does not decrease. Since $SO \leq 2|E|$ and $O(|E|) \in O(m^2)$, the TS-SUBROUTINE converges in $O(m^2)$ time.

⁴Notice that the same proof also implies that if agent 1 **envies** (not necessarily EF1-envies) exactly one agent i and all items in A_i have non-positive value with respect to agent 1, then every other agent has an item with a positive marginal value for agent 1. This generalization helps us prove Theorem 2.

Claim 3. *The TS-SUBROUTINE returns an allocation that is TS. Furthermore, in each iteration of Algorithm 2, the potential function $\Phi = (v(A_1), -n_1)$ does not decrease, and the social welfare $SW(A) = \sum_{i \in N} v(A_i)$ increases by at least 1. Therefore, Algorithm 2 terminates in $O(m^2)$ time.*

Proof. First, we prove that if the subroutine terminates, then the returned allocation is TS. Recall that in a TS allocation, there must be no transfer that decreases the value of either of the agents. Thus, we show that, in the returned allocation, the donator bundle's value reduces during any transfer, thereby proving that the allocation is TS. Notice that, on termination, the while-loop condition ensures that every item is assigned to a bundle where its marginal value is positive. Thus, transferring any item will reduce the value of the donator bundle, proving that the allocation is TS. In the remaining part of the proof we argue that the while loop in the subroutine terminates in polynomial time.

In each iteration of the TS-SUBROUTINE, we identify an item that has been assigned as a strict/weak chore o in some agent i 's bundle. Then, o is reallocated to another (carefully chosen) agent for whom o is a strict good (positive marginal value). Such an agent always exists when $n \geq 3$ by Lemma 2. By the same lemma, when $n \geq 4$, there always exists an agent $k \neq i^*$ (not the special agent) for whom o is a strict good (positive marginal value). We transfer o from a bundle where it is a weak or strict chore (zero or negative marginal value) to one (other than A_{i^*}) where it is a strict good (positive marginal value). This guarantees that no agent's value decreases, and the receiving agent's value strictly increases. Hence, $v(A_1)$ (the minimum value) does not decrease, n_1 (the number of agents that have bundles of value $v(A_1)$) does not increase, and the total social welfare increases by at least 1.

Since $SO \leq 2|E|$ and $O(|E|) \in O(m^2)$, the social welfare can increase at most SO times, i.e. the TS-SUBROUTINE must terminate in $O(m^2)$. $\square$

We are now ready to prove Theorem 2.

Proof of Theorem 2. First, we show that if Algorithm 1 terminates, then the resulting allocation is both EF1 and TS. After that, we prove that the algorithm terminates in polynomial time. By the main while loop in Line 4, the algorithm continues until agent 1 (the agent with the least value) no longer has an EF1 violation towards any other agent. By Lemma 1, this ensures that the allocation is EF1. Next, from Claim 3, we know that the TS-SUBROUTINE guarantees that the resulting allocation is TS. Since this subroutine is called in Lines 8 and 14, the final allocation also satisfies TS.

We now prove that the algorithm terminates in polynomial time. We show that (i) in every iteration of Case I there is a lexicographic improvement in $\Phi = (v(A_1), -n_1)$; and (ii) in Case II, either (a) either Φ is lexicographically improved, or (b) when Φ does not change, the algorithm moves back to Case I. Since each part of the algorithm runs in polynomial time— $v(A_1)$ can increase at most $O(|E|) \in O(m^2)$ times, n_1 can decrease at most $O(n)$ times, the while loop in Case II runs in $O(m)$ time, and the social welfare can improve at most $O(m^2)$ times (which is the potential function for the TS-SUBROUTINE; see Claim 3)—the algorithm terminates in time polynomial in n and m .

Case I. Let i be an agent for whom there is an EF1 violation and let $o \in A_i$ be an item with positive marginal value for A_1 . After reallocating o to agent 1, the value of A_1 strictly increases. In addition, since there was a EF1 violation towards i , we know A_i 's new value is still strictly larger than previous least value i.e. $v(A_i \setminus \{o\}) > v(A_1)$. If $v(A_2) = v(A_1)$, the minimum value does not change, but the number of agents receiving the minimum value n_1 decreases by 1. Otherwise, the minimum value increases by at least 1 and thus Φ lexicographically improves.

Case II. First, we show that in Case II, Φ does not decrease. From Claim 2, we know that any EF1 violation must be towards a *single* special agent i^* and all items in A_{i^*} have a non-positive marginal value for A_1 . In this case, an arbitrary item $o \in A_{i^*}$ is transferred to an agent $k \neq i^*$ who has positive marginal value for o ; the existence of such an agent $k \neq i^*$ is guaranteed via Lemma 2 when $n \geq 4$. Note that $v(A_1)$ does not change since A_1 's bundle is untouched, and no agent becomes worse than $v(A_1)$: Agent i^* is EF1-envied by agent 1 so $v(A_{i^*} \setminus \{o\}) > v(A_1)$, and o has a positive marginal value for A_k .

Thus, for Case II, we need to show that if Φ does not lexicographically increase and the allocation is not EF1, then we must move back to Case I, i.e. there cannot be two consecutive runs of Case II unless the potential function Φ increases. To see this, notice that after the execution of the subroutine in Line 14, if the resulting allocation is not EF1+TS and $v(A_1)$ and n_1 (i.e. Φ) remain unchanged, then the least valued agent remains agent 1. This is because, from Claim 3, we know that the subroutine does not make any agent worse-off. Also, since A_{i^*} was TS before Case II (i.e. contained no non-positive items for itself), it must be the case that i^* was not involved in any transfers during the subroutine execution. Thus, any EF1-violation must be towards a different agent since agent 1 did not EF1-envy i^* after Case II. Moreover, since the least valued agent is still agent 1 and (the new) A_{i^*} is still an envied bundle with no items that have a positive marginal value for A_1 , from Claim 2, the new EF1-envied agent must have an item with positive marginal value for A_1 , satisfying the condition for Case I. $\square$

3.4 Fairness and Efficiency: A Monotonic Result

We have shown that EF1 and the efficiency notion TS exhibits non-monotonic existence. This raises a natural question: *Is there an efficiency notion that is always compatible with EF1, regardless of n ?* We answer this question positively by showing that a slight relaxation of TS—namely WTS—is guaranteed to exist along with EF1 for any n (Theorem 3). Towards this, we provide a polynomial-time algorithm that computes such allocations. We also show that, if we are willing to relax fairness in favor of a stronger efficiency notion, the leximin solution guarantees $\frac{1}{2}$ -EF1 and is PO for any n (Proposition 2). We emphasize that this is only an existence guarantee, as computing an α -EF1 allocation that is also PO is NP-HARD for any α .

3.4.1 Weakening Efficiency

In this section, we show that EF1 is compatible with WTS, a slightly weaker notion than TS, for instances with general graphs and any number of agents.

Theorem 3 (EF1+WTS allocations). *Given a cut-valuation instance, a complete allocation satisfying EF1 and WTS always exists and can be computed in polynomial time.*

Our proof is constructive: we provide an algorithm, akin to Algorithm 1, that computes such an allocation for any instance. The full algorithm and proof of Theorem 3 can be found in Section D. Intuitively, unlike TS which requires each item to have positive marginal in its bundle, WTS permits items to have zero-marginals. This forgoes the need for multiple agents with positive value for an item, removing the need for $n \geq 4$ (Lemma 2).

As a corollary, Theorem 3 implies that any graph with m vertices admits a non-empty n -partition of the vertex set (with $n \leq m$) with nearly-equitable the cut-values across parts. Notably, the bound of Δ in Corollary 1 improves on the previous best of $5\Delta + 1$ by Barman and Verma [2025]. The proof of the Corollary is presented in Section D.

Corollary 1 (Equitable graph cuts). *Given a graph $G = (V, E)$ and an integer $n \leq |V|$, there exists a polynomial-time computation n -partition of V into non-empty $V_1, V_2, \dots, V_n \neq \emptyset$ such that the cut-values satisfy $|v(V_i) - v(V_j)| \leq \Delta$ for all $i, j \in [n]$, where $\Delta := \max_{o \in V} \deg(o)$ denotes the maximum degree of $G = (V, E)$.*

3.4.2 Weakening Fairness

Recall, Theorem 1 shows that, for general graphs, an allocation that is both EF1 and PO need not always exist. In this section, we show that a constant multiplicative approximation of EF1 alongside PO always exists, where a multiplicative approximation of EF1 is defined as follows.

α -EF1. An allocation $A = (A_1, \dots, A_n)$ is said to be α -EF1 if for every $i, j \in N$ such that $v(A_j) > v(A_i)$, there exists an item $o_j \in A_j$ such that $v(A_i) \geq \alpha v(A_j \setminus \{o_j\})$ where $\alpha \in (0, 1]$.

In particular, we show that the leximin solution (defined below) is $\frac{1}{2}$ -EF1.

Leximin An allocation $A = (A_1, \dots, A_n)$ is said to be leximin optimal if A maximizes the value of the agent with the smallest-value, subject to that maximizes the value of the second smallest agent's bundle, and so on.

Proposition 2 (Approximate EF1 and PO). *Given a cut-valuation instance, there always exists an allocation that is $\frac{1}{2}$ -EF1 and PO.*

Note that computing the leximin solution is NP-HARD even for $n = 2$ (same as MAX-CUT). Thus, Proposition 2 is only an existence result; however, efficient computation of α -EF1 and PO allocations is a futile endeavor for any α since finding a PO allocation is NP-hard, even for $n = 2$ (Proposition 1).

Proof of Proposition 2. We show that a leximin solution is $\frac{1}{2}$ -EF1. Since leximin solutions are PO, this completes the proof. For a subset of items $S \subseteq V$ and an item $o \in V$, recall that $N_S(o)$ denotes the number of neighbors of o that belong to S . Thus, the value of any set S , i.e., the number of edges with one endpoint in S and the other in $V \setminus S$, is given by $v(S) = \sum_{o \in S} |N_{V \setminus S}(o)|$.

Let $A = (A_1, \dots, A_n)$ be a leximin allocation, and let i, j be two agents such that i EF1-envies j , i.e., for every $o \in A_j$, we have $v(A_j \setminus \{o\}) > v(A_i)$. Since A is leximin, addition of any item $o \in A_j$ to A_i does not increase the value of A_i , that is, every $o \in A_j$ must have a non-positive marginal value for A_i i.e. $N_{A_i}(o) \geq N_{V \setminus A_i}(o)$. Thus, an upper-bound on the value of A_j can be obtained as

$$\begin{aligned}
v(A_j) &= \sum_{o \in A_j} |N_{V \setminus A_j}(o)| \\
&= \sum_{o \in A_j} \left[|N_{V \setminus A_j}(o) \cap N_{V \setminus A_i}(o)| + |N_{V \setminus A_j}(o) \cap N_{A_i}(o)| \right] \\
&\leq \sum_{o \in A_j} \left[|N_{V \setminus A_j}(o) \cap N_{A_i}(o)| + |N_{V \setminus A_j}(o) \cap N_{A_i}(o)| \right] \\
&= 2 \sum_{o \in A_j} |N_{V \setminus A_j}(o) \cap N_{A_i}(o)| \\
&\leq 2 v(A_i).
\end{aligned}$$

□

4 Forest Graphs

In this section, we provide a polynomial-time algorithm that takes as input a cut-valuation instance where the graph $G = (V, E)$ is a forest (i.e. acyclic), and outputs an allocation that is simultaneously EF1 and socially optimal (SO). This algorithmic guarantee also shows the *existence* of such allocations for forest graphs. Our first observation is a characterization of SO allocations for forests: When $n \geq 2$, no two adjacent items can be allocated to the same agent.

Proposition 3. *Given a cut-valuation instance where G is a forest and $n \geq 2$, an allocation A is SO if and only if, for every $(o, o') \in E$, we have $o \in A_i$ and $o' \in A_j \neq A_i$.*

Proof. A forest is a graph that is acyclic. Since a graph is bipartite if and only if it contains no cycles of odd length, a forest must be bipartite. Let the bipartite graph corresponding to the forest be $G = (V_1 \sqcup V_2, E)$. Consider the social welfare of the allocation $A = (V_1, V_2, \emptyset, \dots, \emptyset)$.

$$\text{SW}(A) = \sum_{i \in N} v(A_i) = v(V_1) + v(V_2) = 2|E|.$$

In any graph, since each edge can at most contribute to the valuation of both its endpoints, the optimal social welfare is bounded above by $2|E|$, i.e., $\text{SO} \leq 2|E|$. Thus, the optimal welfare for forest graphs is exactly $2|E|$, which means that no adjacent items can be allocated to the same agent in any SO allocation. $\square$

As consequence of Proposition 3, the problem of finding an allocation that is EF1 and SO is equivalent to finding an EF1 allocation where no adjacent vertices (items) are assigned to the same agent. Under this condition, agent's bundle forms an independent set S in the graph, with $v(S) = \sum_{o \in S} \deg(o)$. This coincides with the conflict-constraints, albeit for valuations derived from cut-functions.

Note that for $n = 2$, the bi-partition based on the bipartite graph of the forest is both EF and SO since the cut function is symmetric, i.e. $v(S) = v(V \setminus S)$ for any $S \subseteq V$ ⁵; thus we assume $n \geq 3$. Interestingly, the negative example in Theorem 1 is also a bipartite graph (with treewidth = 2), but an EF1+SO allocation does not exist for it when $n = 3$. This highlights that while strong existence and algorithmic guarantees (EF1+SO) hold for forests (bipartite; treewidth 1), they break even for graphs that are only slightly different.

Algorithm Overview. Algorithm 3 begins with an empty initial allocation and an arbitrary rooting of each tree in the forest. The algorithm proceeds iteratively: at each step, it assigns a root item (vertex) to an agent and roots the resulting subtrees at the unallocated neighbors (children) of the assigned item. Since we always exclusively assign root items, at most one of their neighbors—the parent—may have been already assigned. Therefore, to satisfy the SO condition, it suffices to ensure that the assigned root item is not in the same bundle as its parent. We refer to such a root as a *feasible root* (for a bundle)—i.e. one whose parent does not belong to that bundle.

Throughout the algorithm, we relabel bundles to ensure that $v(A_1) \leq v(A_2) \leq \dots \leq v(A_n)$ holds at all times (Lemma 1). Whenever there exists a feasible root for the least valued agent, aka agent 1, the algorithm assigns this root to A_1 as part of CASE 1. CASES 2 and 3 handle situations where no root item is feasible for A_1 . Here, the algorithm assigns a root item to a different agent, temporarily violating the EF1 condition. However, it is carefully designed to *catch up* on such EF1 violation through subsequent assignments by assigning o_t 's children to agent 1 (or others as

⁵In fact, this is true in more generality for *any* bipartite graph (not just forests) and $n = 2$ agents.

needed). While doing so, we first prioritize the **leaf-children** of a node o_t , i.e. the children of o_t (unallocated neighbors) that have a degree of 1. This helps us ensure that, as long as there is an unallocated item, at least one of the three CASES must hold.

Algorithm 3 Rooting peeling algorithm for EF1 + SO allocations on forests

```

1: Initialize with an empty allocation  $A = (\emptyset, \dots, \emptyset)$ .
2: Root each of the  $k$  trees at arbitrary nodes  $r_1, \dots, r_k$ .
3: while (there exists an unallocated item) do
4:   Relabel bundles s.t.  $v(A_1) \leq v(A_2) \leq \dots \leq v(A_n)$ .



CASE 1: The minimum-valued agent has a feasible root.


5: if (agent 1 has a feasible root node  $o_t$ ) then
6:   Allocate  $o_t$  to agent 1, i.e.,  $A_1 \leftarrow A_1 \cup \{o_t\}$ .
7:   while (there is an unallocated leaf-child  $o_\ell$  of  $o_t$ ) do
8:     Let  $F = \arg \min_{i \in [2, n]} v(A_i)$ , and consider an agent  $j \in F$ . If  $|F| \geq 2$ , choose  $j \in F$  such that
       there exists an agent in  $F \setminus \{j\}$  with a feasible root.
9:     Allocate  $o_\ell$  to  $j$ , i.e.,  $A_j \leftarrow A_j \cup \{o_\ell\}$ .
10:  end while

CASE 2: A feasible root can be allocated to the second minimum-valued agent.


11: else if ( $v(A_1) > v(A_2 \setminus \{o_2^*\})$  or agent 2 has a feasible root  $o_t$  such that  $v(o_t) > v(o_2^*)$ ) then
12:   Allocate  $o_t$  to agent 2, i.e.,  $A_2 \leftarrow A_2 \cup \{o_t\}$ .
13:   Let  $h_2^* \in A_2$  be s.t.  $h_2^* \in \arg \min_{h \in A_2} v(A_2 \setminus \{h\})$ .
14:   while (there is an unallocated leaf-child  $o_\ell$  of  $o_t$ ) do
15:     Let  $F = \arg \min_{i \in [1] \cup [3, n]} v(A_i)$ , and consider an agent  $j \in F$ . If  $|F| \geq 2$ , choose  $j \in F$  such that
       there exists an agent in  $F \setminus \{j\}$  with a feasible root.
16:     Allocate  $o_\ell$  to  $j$ , i.e.,  $A_j \leftarrow A_j \cup \{o_\ell\}$ .
17:   end while
18:   while ( $v(A_1) < v(A_2 \setminus \{h_2^*\})$ ) do
19:     Allocate an unallocated child of  $o_t$  to agent 1.
20:   end while

CASE 3: A feasible root can be allocated to some other agent.


21: else if (there exists  $j \in \arg \min_{i \in [3, n]} v(A_i \setminus \{o_i^*\})$  such that  $v(A_1) > v(A_j \setminus \{o_j^*\})$ ) then
22:   Allocate  $o_t$  to agent  $j$ , i.e.,  $A_j \leftarrow A_j \cup \{o_t\}$ .
23:   Allocate all leaves of  $o_t$  to agent 1.
24:   while ( $v(A_1) < \min\{v(A_2), v(A_j \setminus \{o_j^*\})\}$ ) do
25:     Allocate an unallocated child of  $o_t$  to agent 1.
26:   end while
27: end if
28: end while

```

In particular, the algorithm proceeds by checking for the following three conditions sequentially:

- CASE 1: *Is there a feasible root o_t for the least-valued agent (agent 1)?*
- CASE 2: *Can we allocate a feasible root o_t to the second least-valued agent (agent 2) such that any resulting EF1 violation can be compensated for?*
- CASE 3: *Is there an agent $j \neq 1, 2$ who can receive a feasible root o_t such that any resulting EF1 violation can be remedied?*

Throughout, we use o_i^* to denote the item whose removal from A_i causes the maximum drop in value of A_i , i.e., $o_i^* \in \arg \min_{o \in A_i} v(A_i \setminus \{o\})$.

Theorem 4 (Existence of EF1 + SO allocations). *Given cut-valuation instance where $G = (V, E)$ is a forest, a complete allocation that is EF1 and SO always exists and can be computed in polynomial time.*

The algorithm, by construction, always assigns feasible root items, thereby ensuring that no two adjacent items are allocated to the same agent, i.e., the returned allocation is SO. For EF1, assigning an item to agent 1 in CASE 1 preserves EF1, since on removal of the last added item, A_1 was the least-valued bundle. In contrast, when an item o_t is assigned to an agent $j \neq 1$, each child of o_t (unallocated neighbor of o_t) becomes a feasible root item for agent 1, where each child has a value of at least 1. Note that o_t has at least $v(\{o_t\}) - 1$ (unassigned) children (since at most one neighbor of o_t , i.e. its parent has been assigned so far). Thus, agent 1 can be *almost* compensated for the resulting EF1-violation (of at most $v(\{o_t\})$ toward j ! CASES 2 and 3 capture cases where this EF1-violation can be *fully* eliminated. In particular, Lemma 3 shows that, after every execution of each of the CASES, the EF1 property is maintained.

Lemma 3. *Consider an iteration of Algorithm 3 in which CASE X is executed (where $X \in [3]$), and let the (partial) allocation before the iteration be A^{old} and after the iteration be A^{new} . If A^{old} is EF1, then A^{new} is also EF1.*

Since the algorithm starts with an empty allocation which is trivially EF1, inductively one can argue that the final resulting allocation will also be EF1. It remains to show that at least one of the three CASES is always satisfied whenever an item remains to be allocated [Lemma 4]. In Lemma 4, we show that Algorithm 3 always terminates with a complete allocation; that is, as long as there is an unallocated item, it can never be the case that none of the if-conditions (CASES) is satisfied.

Lemma 4. *During the execution of the algorithm whenever there is an unallocated item, at least one of the three CASES must hold.*

Combining these lemmas, we get that Algorithm 3 returns a complete allocation (by Lemma 4) in $O(m)$ time that is EF1 (by Lemma 3) and SO (by construction). Thus, Theorem 4 stands proved. The proofs of these lemmas are provided in Section 4.1.

4.1 Proof of Theorem 4

As discussed, our algorithm, by construction ensures that the returned allocation is SO—no adjacent vertices (items) are allocated to the same agent. Thus, in order to prove Theorem 4, we need to show that the Algorithm 3 returns a complete allocation (Lemma 4) that is also EF1 (Lemma 3). Before we prove these lemmas, we make a useful observation: At any iteration of the algorithm, no unallocated root node is leaf since all leaf-children are assigned in the same iteration as its parent. This is by construction—Lines 7 to 10, 14 to 17, and 23 in Algorithm 3 ensure that all leaf-children of a root node o_t are assigned in the same iteration as o_t .

Observation 1. *During any iteration of the algorithm, no unallocated root node is a leaf.*

Throughout this section, we assume that $n \geq 3$ and that there are no isolated nodes (i.e. all nodes have degree at least 1). With this, we are now ready to prove the two lemmas. We begin by proving that the returned allocation is EF1. Toward this, we prove a stronger claim — in any iteration of the algorithm, if the (partial) allocation before the iteration was EF1, then the (partial) allocation after this iteration will also be EF1. Since the algorithm starts with an empty allocation that is trivially EF1, inductively one can argue that the final resulting allocation will also be EF1.

Lemma 3. Consider an iteration of Algorithm 3 in which CASE X is executed (where $X \in [3]$), and let the (partial) allocation before the iteration be A^{old} and after the iteration be A^{new} . If A^{old} is EF1, then A^{new} is also EF1.

Proof. Since the algorithm relabels the bundles to maintain non-decreasing bundle valued from A_1 to A_n , we have

$$v(A_1^{\text{old}}) \leq v(A_2^{\text{old}}) \leq \dots \leq v(A_n^{\text{old}}) \quad (2)$$

Furthermore, let $o_i^* \in \arg \min_{o \in A_i^{\text{old}}} v(A_i^{\text{old}} \setminus \{o\})$ and similarly $h_i^* \in \arg \min_{h \in A_i^{\text{new}}} v(A_i^{\text{new}} \setminus \{h\})$.

Our analysis is based on which case $X \in [3]$ is executed. For each of these, we show that there is no EF1 violation between any pair of agents. We start by considering that CASE 1 was executed in going from A^{old} to A^{new} .

CASE 1: The assignment in CASE 1 of Algorithm 3 is such that agent 1 receives o_t and other agents receive the leaf-children of o_t , if any. That is, no agent's bundle value goes down. Consider $i, j \in [2, n]$.

From $i \rightarrow 1$: First, we show that there is no EF1-envy from an agent $i \in [2, n]$ towards agent 1. Since in the value of i 's bundle does not decrease (i may or may not receive an additional leaf-child of o_t), we have $v(A_i^{\text{new}}) \geq v(A_i^{\text{old}}) \geq v(A_1^{\text{old}}) = v(A_1^{\text{new}} \setminus \{o_t\})$. The second inequality follows from (2), and the third inequality is due to the update $A_1^{\text{new}} = A_1^{\text{old}} \cup \{o_t\}$. Thus, no agent has an EF1 violation towards agent 1.

From agent 1 $\rightarrow i$: Next, we show that there is no EF1 violation from agent 1 to any other agent $i \in [2, n]$. Since A^{old} was EF1, we know that $v(A_1^{\text{old}}) \geq v(A_i^{\text{old}} \setminus \{o_i^*\})$. Also, since a node cannot have more children than its own value (which is equal to its degree), we have $v(o_t) = \deg(o_t) \geq \# \text{ leaf-children of } o_t$. Thus, even if i receives *all* of o_t 's leaf-children, we get,

$$v(A_1^{\text{new}}) = v(A_1^{\text{old}}) + v(o_t) \geq v(A_i^{\text{old}} \setminus \{o_i^*\}) + \# \text{ leaf-children of } o_t \geq v(A_i^{\text{new}} \setminus \{o_i^*\}).$$

That is, agent 1 does not EF1-envy any agent $i \in [2, n]$.

From $i \rightarrow j$: Finally, consider any $i, j \in [2, n]$ such that $v(A_j^{\text{new}}) > v(A_i^{\text{new}})$. If j does not receive any item in this iteration, since A^{old} was EF1, we get

$$v(A_i^{\text{new}}) \geq v(A_i^{\text{old}}) \geq v(A_j^{\text{old}} \setminus \{o_j^*\}) = v(A_j^{\text{new}} \setminus \{o_j^*\}),$$

implying that i does not EF1-envy j . On the other hand, if j receives an item in this iteration, then by Line 8 in Algorithm 3, we have that $v(A_j^{\text{new}}) = v(A_j^{\text{old}}) + 1$ since only leaf-children, whose value is 1, are allocated to the least-bundle in $[2, n]$. Thus, removal of any item from A_j eliminates this envy since the graph contains no isolated vertices with value 0.

CASE 2: The assignment is CASE 2 of is such that agent 2 receives a root node o_t , and the children of o_t , starting with the leaf-children, are assigned to the agent in $1 \cup [3, n]$ with the least value. Consider $i, j \in [3, n]$.

From $i \rightarrow 2$: For any agent $i \in [3, n]$ to agent 2, notice from (2), that $v(A_2^{\text{old}}) \leq v(A_i^{\text{old}})$. Thus, we have $v(A_i^{\text{new}}) \geq v(A_i^{\text{old}}) \geq v(A_2^{\text{old}}) = v(A_2^{\text{new}} \setminus \{o_t\})$, where the final inequality is because of the update $A_2^{\text{new}} = A_2^{\text{old}} \cup \{o_t\}$. Thus, no agent $i \in [3, n]$ EF1-envies agent 2 in A^{new} .

From agent 2 $\rightarrow i$: Notice that, in A^{new} , agent 2 (who receives o_t) does not EF1-envy any other agent $i \in [3, n]$ (who may receive some leaf-children of o_t). This is because A^{old} was EF1, and $v(o_t) \geq \# \text{ leaf-children of } o_t$, whereby

$$v(A_2^{\text{new}}) = v(A_2^{\text{old}}) + v(o_t) \geq v(A_i^{\text{old}} \setminus \{o_i^*\}) + \# \text{ leaf-children of } o_t \geq v(A_i^{\text{new}} \setminus \{o_i^*\}).$$

From $i \rightarrow j$: If $v(A_j^{\text{new}}) > v(A_i^{\text{new}})$ and j does not receive any item in this iteration (i.e. $A^{\text{new}} = A^{\text{old}}$), then $v(A_i^{\text{new}}) \geq v(A_i^{\text{old}}) \geq v(A_j^{\text{old}} \setminus \{o_j^*\}) = v(A_j^{\text{new}} \setminus \{o_j^*\})$. That is, there is no EF1-envy from i to j . If j receives an item in this iteration, then by Line 15 in Algorithm 3, we have that $v(A_j^{\text{new}}) = v(A_i^{\text{new}}) + 1$ (value of a leaf is 1), implying that the removal of any item from A_j eliminates this envy.

From agent 1 $\rightarrow i$ and 1 $\rightarrow 2$: If some agent $i \in [3, n]$ also receives an item, we know that $v(A_1^{\text{new}}) \geq v(A_i^{\text{old}}) \geq v(A_2^{\text{old}}) = v(A_2^{\text{new}} \setminus \{o_t\})$. Towards agent $j \in [3, n]$ if j does not receive an item, we know that $v(A_1^{\text{new}}) > v(A_1^{\text{old}}) \geq v(A_j^{\text{old}} \setminus \{o_j^*\}) = v(A_j^{\text{new}} \setminus \{o_j^*\})$. If j receives an item, then by Line 15, we know $v(A_j^{\text{new}}) \leq v(A_1^{\text{new}}) + 1$; thus, on removal of any item from j 's bundle, any envy will be eliminated.

If no agent in $i \in [3, n]$ receives an item: Agent 1 does not EF1-envy i since A^{old} is EF1, and $v(A_1^{\text{new}}) \geq v(A_1^{\text{old}})$. Towards agent 2, by the while condition in Line 18, we know that when the while loop terminates, there is no EF1-envy from agent 1 to 2. It remains to show that the while loop will terminate, i.e., there are enough children of o_t to allocate to agent 1 such that $v(A_1^{\text{new}})$ can reach $v(A_2^{\text{new}} \setminus \{h_2^*\})$. Since o_t has $v(o_t) - 1$ children, each of which contributes at least 1 to agent 1's bundle when allocated, giving o_t 's children to agent 1 can increase her value by at least $v(o_t) - 1$. If $v(A_1^{\text{old}}) \geq v(A_2^{\text{old}}) - v(o_2^*) + 1$, we have

$$\begin{aligned} v(A_1^{\text{old}}) + v(o_t) - 1 &\geq v(A_2^{\text{old}}) - v(o_2^*) + v(o_t) \\ &= v(A_2^{\text{new}} \setminus v(o_2^*)) \geq v(A_2^{\text{new}} \setminus v(h_2^*)). \end{aligned}$$

On the other hand, if $v(o_t) \geq v(o_2^*) + 1$, we know that $h_2^* = o_t$. Then,

$$\begin{aligned} v(A_1^{\text{old}}) + v(o_t) - 1 &\geq v(A_2^{\text{old}}) - v(o_2^*) + v(o_t) - 1 \\ &\geq v(A_2^{\text{old}}) = v(A_2^{\text{new}} \setminus v(h_2^*)). \end{aligned}$$

From $i \rightarrow 1$ and 2 $\rightarrow 1$: With this, we are only left to evaluate the possible envy from agent $i \in [2, n]$ to agent 1. For this, we perform the analysis by considering two cases based on whether the last allocated item of this iteration (o_f) was done in Line 16 or in Line 19.

If the last allocated item was in Line 16: If $v(A_1^{\text{new}}) > v(A_i^{\text{new}})$, then from Line 15, it must be the case that $v(A_1^{\text{new}}) = v(A_i^{\text{new}}) + 1$, whereby any envy can be eliminated by removing any item from agent 1's bundle.

If the last allocated item was in Line 19: From the while condition in Line 19, we know that $v(A_1^{\text{new}} \setminus \{o_f\}) < v(A_2^{\text{new}} \setminus \{h_2^*\})$. Furthermore, we know from the allocation process that $A_2^{\text{new}} = A_2^{\text{old}} \cup \{o_t\}$ whereby $v(A_2^{\text{new}} \setminus \{o_t\}) = v(A_2^{\text{old}})$. Finally, from (2), we know that $v(A_2^{\text{old}}) \leq v(A_i^{\text{old}}) \leq v(A_i^{\text{new}})$; the second inequality is because no agent's bundle decreases in value. Thus, combining all, we have

$$v(A_1^{\text{new}} \setminus \{o_f\}) < v(A_2^{\text{new}} \setminus \{h_2^*\}) \leq v(A_2^{\text{new}} \setminus \{o_t\}) = v(A_2^{\text{old}}) \leq v(A_i^{\text{old}}) \leq v(A_i^{\text{new}}).$$

That is, no agent $i \in [2, n]$ EF1-envies agent 1.

With this, we consider the final case and show that if CASE 3 was executed while going from $A^{\text{old}} \rightarrow A^{\text{new}}$ and if A^{old} was EF1, then A^{new} must also be EF1.

CASE 3: Let $j \in [3, n]$ be the agent who receives the feasible root o_t . Also, consider an arbitrary agent $i \in [2, n]$ such that $i \neq j$. Notice that any such agent i is not allocated an item in this iteration.

From any agent $k \neq i$ to i : For any i , notice that $v(A_i^{\text{new}}) = v(A_i^{\text{old}})$. Also, for every agent k , we have, $v(A_k^{\text{new}}) \geq v(A_k^{\text{old}})$. Thus, since A^{old} was EF1, we get that

$$v(A_k^{\text{new}}) \geq v(A_k^{\text{old}}) \geq v(A_i^{\text{old}} \setminus \{o_i^*\}) = v(A_i^{\text{new}} \setminus \{o_i^*\})$$

From $i \rightarrow j$: Since CASE 2 does not hold but A^{old} is EF1, we get that $v(A_1^{\text{old}}) = v(A_2^{\text{old}} \setminus \{o_2^*\})$. Thus, from the condition of CASE 3, we get $v(A_j^{\text{old}}) - v(o_j^*) < v(A_1^{\text{old}}) = v(A_2^{\text{old}}) - v(o_2^*)$. Adding $v(o_t)$ to both sides, we get

$$v(A_j^{\text{old}}) - v(o_j^*) + v(o_t) \leq v(A_2^{\text{old}}) - v(o_2^*) + v(o_t) \leq v(A_2^{\text{old}}),$$

where the last inequality is because CASE 2 does not hold and it must be the case that $v(o_t) \leq v(o_2^*)$. Moreover, the left side of the equation is simply $v(A_j^{\text{new}}) - v(o_j^*)$ since o_t was allocated to agent j . Also, from (2) and the values of bundles do not decrease, we get that $v(A_2^{\text{old}}) \leq v(A_i^{\text{old}}) \leq v(A_i^{\text{new}})$ for agent $i \in [2, n]$. Combining it all together, we get

$$v(A_j^{\text{new}}) - v(o_j^*) \leq v(A_i^{\text{new}}),$$

i.e., there is no EF1-envy from any agent $i \in [2, n]$ to j .

From agent 1 $\rightarrow j$: To show that there is no EF1-envy toward j , we need to show that the while loop through Lines 24 to 26 terminates. That is, $v(A_1^{\text{new}}) \geq v(A_2^{\text{old}})$ or $v(A_1^{\text{new}}) \geq v(A_j^{\text{new}} \setminus \{o_j^*\})$. Note that we have just shown that there is no EF1-envy towards j if the former holds. In particular, we need to show that there are enough children of o_t to cover-up for any created EF1-envy toward j . For this, notice that, from the condition of CASE 3, we have that $v(A_j^{\text{old}}) - v(o_j^*) < v(A_1^{\text{old}})$, i.e.,

$$v(A_j^{\text{old}}) - v(o_j^*) + 1 \leq v(A_1^{\text{old}}).$$

Moreover, we know that $\#$ children of $o_t \in \{v(o_t), v(o_t) - 1\}$, i.e.,

$$v(o_t) - 1 \leq \# \text{ of children of } o_t.$$

Thus, adding the two equations, we get

$$v(A_j^{\text{new}} \setminus \{o_j^*\}) = v(A_j^{\text{old}}) - v(o_j^*) + v(o_t) \leq v(A_1^{\text{old}}) + (\# \text{ of children of } o_t).$$

From any agent $k \rightarrow 1$: First, we claim that there is no envy toward agent 1 after all leaf-children of o_t are assigned to agent 1's bundle. Note that since we are in CASE 3, it means that CASE 2 does not hold. That is, it must be the case that $v(o_t) \leq v(o_2^*)$ and $v(A_1^{\text{old}}) = v(A_2^{\text{old}} \setminus \{o_2^*\})$ since A^{old} is EF1. Using this and the fact that the $\#$ leaf-children of $o_t \leq v(o_t)$, we get

$$v(A_1^{\text{old}}) + \# \text{ leaf-children of } o_t \leq v(A_1^{\text{old}}) + v(o_t) = v(A_2^{\text{old}} \setminus \{o_2^*\}) + v(o_t) \leq v(A_2^{\text{old}}).$$

From (2), we know that $v(A_2^{\text{old}}) \leq v(A_k^{\text{old}})$ and since the bundle values do not decrease, we get $v(A_1^{\text{old}}) + \# \text{ leaf-children of } o_t \leq v(A_k^{\text{new}})$.

If the last allocated item was in Line 25, from its while loop condition we know that before that last item o_f was allocated to agent 1, we had $v(A_1^{\text{new}} \setminus \{o_f\}) < v(A_2^{\text{old}}) \leq v(A_k^{\text{new}})$. Thus, in A^{new} , any agent k does not envy agent 1 up to the removal of the last added item o_f . $\square$

Lemma 4. *During the execution of the algorithm whenever there is an unallocated item, at least one of the three CASES must hold.*

Proof. Toward a contradiction, suppose that we encounter a point for the first time when none of the three if-conditions (CASES) are satisfied. Since this is the first time such a situation occurs, it must be that one of the if-conditions was satisfied in the previous iteration. Let the allocation before that iteration be $A^{\text{old}} = (A_1^{\text{old}}, \dots, A_n^{\text{old}})$ and the allocation after that iteration be $A^{\text{new}} = (A_1^{\text{new}}, \dots, A_n^{\text{new}})$. Let $o_i^* \in \arg \min_{o \in A_i^{\text{old}}} v(A_i^{\text{old}} \setminus \{o\})$ and similarly $h_i^* \in \arg \min_{h \in A_i^{\text{new}}} v(A_i^{\text{new}} \setminus \{h\})$. Again, without loss of generality, we assume that $v(A_1^{\text{old}}) \leq v(A_2^{\text{old}}) \leq \dots \leq v(A_n^{\text{old}})$.

We will show that A^{new} must satisfy at least one of the three CASES. In particular, we consider three cases based on whether the first, second, or third if-loop was executed when going from A^{old} to A^{new} .

CASE 1: The first if-loop was executed (i.e., agent 1 received a feasible root o_t).

If agent 1 is still the minimum-valued agent, we argue that either CASE 2 OR 3 must hold, i.e., there exists an agent j for whom $v(A_1^{\text{new}}) > v(A_j^{\text{new}}) - v(h_j^*)$. We know that $v(o_t)$ is at least as much as the number of leaf-children of o_t , i.e. $v(o_t) \geq \# \text{ leaf-children of } o_t$. Since $n \geq 3$, there exists an agent j who does not receive all leaf-children of o_t , i.e., $v(o_t) > \# \text{ leaf-children of } o_t \text{ received by } j$. Also, from Lemma 3, we know that the allocation A^{old} is EF1, i.e., $v(A_1^{\text{old}}) \geq v(A_j^{\text{old}} \setminus \{o_j^*\})$. Combining both together, we get that

$$\begin{aligned} v(A_1^{\text{new}}) &= v(A_1^{\text{old}}) + v(o_t) \\ &> v(A_j^{\text{old}} \setminus \{o_j^*\}) + \# \text{ leaf-children of } o_t \text{ received by } j \\ &= v(A_j^{\text{new}} \setminus \{o_j^*\}) \geq v(A_j^{\text{new}} \setminus \{h_j^*\}). \end{aligned}$$

If agent 1 is no longer the least-valued agent, let the least-valued agent be k . If k received a leaf-child of o_t , we know that $v(A_k^{\text{new}}) > v(A_k^{\text{old}})$. Since agent 1 was the least-valued agent in A^{old} , $v(A_k^{\text{old}}) \geq v(A_1^{\text{old}})$. Moreover, $v(A_1^{\text{old}}) = v(A_1^{\text{new}}) - v(o_t)$. Combining all, we get

$$v(A_k^{\text{new}}) > v(A_1^{\text{new}}) - v(o_t) \geq v(A_1^{\text{new}} \setminus \{h_1^*\}),$$

i.e., there exists an agent k against whom CASE 2 OR 3 must hold.

Finally, if agent k did not receive any leaf-children of o_t and k is the minimum-valued agent, then if o_t has any non-leaf children, then such a node would be a feasible root node for k , i.e., we would be in CASE 1. If o_t 's children are only leaf nodes, then, when the last leaf-child of o_t was being added, there must have been two nodes with the same minimum value. In this case, Line 8 in Algorithm 3 ensures that k has a feasible root, whereby again satisfying in CASE 1.

From Observation 1, these are the only possibilities since o_t cannot be a leaf.

CASE 2: The second if-loop was executed (i.e., the second-minimum valued agent 2 received a feasible root o_t).

If the while-loop through Lines 18 to 20 does not get executed, or, if there is a time when $v(A_1^{\text{new}}) = v(A_3)$ during execution of Lines 14-17, then the condition in Line 15 ensures that for an agent with the least value, there must be a feasible root, i.e., CASE 1 must hold. If $v(A_1^{\text{new}})$ never reaches $v(A_3)$, then only agent 1 may get a leaf-child of o_t during execution of Lines 14-17. Since $v(A_1^{\text{old}}) < v(A_2^{\text{old}})$ and $\# \text{ leaf-children of } o_t \leq v(o_t)$, we have $v(A_1^{\text{new}}) < v(A_2^{\text{new}})$ and thus agent 1 is still the least agent. Consider two possibilities: First, if agent 1 does not get a leaf-child of o_t either, then all children of o_t are non-leaf nodes and become feasible roots for agent 1, i.e., CASE 1 must hold. Second, if agent 1 gets o_t 's leaf children, then $v(A_1^{\text{new}}) > v(A_1^{\text{old}}) \geq v(A_i^{\text{old}} \setminus \{o_i^*\}) = v(A_i^{\text{new}} \setminus \{h_i^*\})$ holds for every $i \in [3, n]$, i.e., CASE 2 OR 3 must hold.

If the while-loop through Lines 18 to 20 gets executed, then just before the while-loop, it holds that

$$\begin{aligned} v(A_1^{\text{new}}) &< v(A_2^{\text{new}} \setminus \{h_2^*\}) \\ &\leq v(A_2^{\text{new}} \setminus \{o_t\}) \\ &= v(A_2^{\text{old}}) \leq v(A_3^{\text{old}}). \end{aligned}$$

This implies that no agent $i \in [3, n]$ obtains an item during execution of Lines 14-17. If $i \in [2, n]$ is the least valued agent in A^{new} , we show that we are in CASE 2 OR 3. Before the last item o_f was allocated to A_1 , we know

$$\begin{aligned} v(A_1^{\text{new}} \setminus \{o_f\}) &< v(A_2^{\text{new}} \setminus \{h_2^*\}) \\ &\leq v(A_2^{\text{new}} \setminus \{o_t\}) \\ &\leq v(A_i^{\text{old}}) \leq v(A_i^{\text{new}}). \end{aligned}$$

If agent 1 remains the least-valued agent in A^{new} , we have

$$v(A_i^{\text{new}} \setminus \{h_i^*\}) = v(A_i^{\text{old}} \setminus \{o_i^*\}) \leq v(A_1^{\text{old}}) < v(A_1^{\text{new}}),$$

where the second last inequality is because A^{old} satisfies EF1 and the last inequality is because agent 1 gets items during execution of the while-loop through Lines 18 to 20. Therefore, CASE 2 OR 3 must hold for the least-valued agent in A^{new} .

CASE 3: The third if-loop was executed (i.e., a feasible root was allocated to the agent $j \in [3, n]$ with the minimum $v(A_j \setminus \{o_j^*\})$).

If an agent $i \in [2, n]$ is the least-valued agent in A^{new} , recall that after allocating all o_t 's leaf-children to agent 1, we have $v(A_1^{\text{new}}) = v(A_1^{\text{old}}) + \# \text{ leaf-children of } o_t \leq v(A_2^{\text{old}})$. Combining this fact with the while condition on Line 24, we know that before the final item, call o_f , was allocated to agent 1 ,

$$v(A_1^{\text{new}} \setminus \{o_f\}) < v(A_2^{\text{old}}) \leq v(A_i^{\text{old}}) \leq v(A_i^{\text{new}}).$$

Thus, either CASE 2 OR 3 must hold for the least-valued agent i in A^{new} .

If agent 1 remains the least-valued agent, we note that the bundle of any agent in $i \in [2, n] \setminus \{j\}$ has not changed. If agent 1 receives an item, then we have

$$v(A_1^{\text{new}}) > v(A_1^{\text{old}}) \geq v(A_i^{\text{old}} \setminus \{o_i^*\}) = v(A_i^{\text{new}} \setminus \{h_i^*\}),$$

which means that CASE 2 OR 3 must hold for agent 1 (minimum-valued agent in A^{new}). If agent 1 does not receives an item either, then all children of o_t are non-leaf nodes and become feasible roots for agent 1, i.e., CASE 1 must hold. $\square$

5 Concluding Remarks

We studied the compatibility of EF1 with various efficiency guarantees under cut-valuations, noting surprising non-monotonicity in existence guarantees against the number of agents. Since the cut-valuations are identical, all our results extend to the equitability (EQ)-based fairness notions. Future work includes exploring stronger fairness notions such as EFX and EQX. Another direction is to relax the fairness requirement. In Section 3.4.2, we show that a $\frac{1}{2}$ -EF1 allocation that is also PO always exists for general graphs. While the approximation factor cannot improve beyond $\frac{2}{3}$ (Theorem 1), it remains an open question whether (and how far) it can be improved. Finally, understanding the complexity of deciding whether a given instance admits an EF1 allocation satisfying a target efficiency criterion $X \in \{\text{TS}, \text{PO}, \text{SO}\}$ is an interesting future direction.

Acknowledgments

This research was supported by the National Science Foundation (NSF) through CAREER Award IIS-2144413 and Award IIS-2107173. Yu Zhou is supported by Research Start-up Funds of Beijing Normal University at Zhuhai (No. 310425209505). We thank the anonymous reviewers for their fruitful comments. We are grateful to Siddharth Barman for introducing this problem, and for many helpful discussions.

References

- 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.
- Siddharth Barman and Paritosh Verma. Existence and computation of maximin fair allocations under matroid-rank valuations. *arXiv preprint arXiv:2012.12710*, 2020.
- Siddharth Barman and Paritosh Verma. Fair division beyond monotone valuations. *arXiv preprint arXiv:2501.14609*, 2025.
- Siddharth Barman, Vishwa Prakash HV, Aditi Sethia, and Mashbat Suzuki. Fair and efficient allocation of indivisible mixed manna. *arXiv preprint arXiv:2507.03946*, 2025.
- Xiaohui Bei, Ayumi Igarashi, Xinhang Lu, and Warut Suksompong. The price of connectivity in fair division. *SIAM journal on Discrete Mathematics*, 36(2):1156–1186, 2022.
- Umang Bhaskar, AR Sricharan, and Rohit Vaish. On approximate envy-freeness for indivisible chores and mixed resources. *arXiv preprint arXiv:2012.06788*, 2020.
- Umang Bhaskar, Gunjan Kumar, Yeshwant Pandit, and Rakshitha. Towards envy-freeness relaxations for general nonmonotone valuations. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pages 298–306, 2025.
- 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 and Economic Behavior*, 131:197–221, 2022.
- Vittorio Bilò, Martin Loebl, and Cosimo Vinci. On almost fair and equitable allocations of indivisible items for non-monotone valuations. *arXiv preprint arXiv:2503.05695*, 2025.

- Arpita Biswas and Siddharth Barman. Fair division under cardinality constraints. In *IJCAI*, pages 91–97. ijcai.org, 2018.
- Arpita Biswas and Siddharth Barman. Matroid constrained fair allocation problem. In *AAAI*, pages 9921–9922. AAAI Press, 2019.
- Sylvain Bouveret, Katarína Cechlárová, Edith Elkind, Ayumi Igarashi, and Dominik Peters. Fair division of a graph. In *IJCAI*, pages 135–141. ijcai.org, 2017.
- Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.
- Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, 7(3):1–32, 2019.
- Nina Chiarelli, Matjaž Krnc, Martin Milanič, Ulrich Pferschy, Nevena Pivač, and Joachim Schauer. Fair packing of independent sets. In *Combinatorial Algorithms: 31st International Workshop, IWOCA 2020, Bordeaux, France, June 8–10, 2020, Proceedings 31*, pages 154–165. Springer, 2020.
- George Christodoulou and Vasilis Christoforidis. Fair and truthful allocations under leveled valuations. *arXiv preprint arXiv:2407.05891*, 2024.
- George Christodoulou, Amos Fiat, Elias Koutsoupias, and Alkmini Sgouritsa. Fair allocation in graphs. In *Proceedings of the 24th ACM Conference on Economics and Computation*, pages 473–488, 2023.
- Benjamin Cookson, Soroush Ebadian, and Nisarg Shah. Constrained fair and efficient allocations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 13718–13726, 2025.
- Argyrios Deligkas, Eduard Eiben, Tiger-Lily Goldsmith, and Viktoriia Korchemna. Ef1 and efx orientations. *arXiv preprint arXiv:2409.13616*, 2024.
- Argyrios Deligkas, Eduard Eiben, Stavros D Ioannidis, Dušan Knop, and Šimon Schierreich. Balanced and fair partitioning of friends. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 13754–13762, 2025.
- Michael R Garey, David S Johnson, and Larry Stockmeyer. Some simplified np-complete problems. In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 47–63, 1974.
- Hadi Hosseini and Šimon Schierreich. The algorithmic landscape of fair and efficient distribution of delivery orders in the gig economy. *arXiv preprint arXiv:2503.16002*, 2025.
- Hadi Hosseini, Aghaheybat Mammadov, and Tomasz Wąs. Fairly allocating goods and (terrible) chores. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 2738–2746, 2023.
- Hadi Hosseini, Shivika Narang, and Tomasz Wąs. Fair distribution of delivery orders. *Artificial Intelligence*, page 104389, 2025.
- Halvard Hummel and Magnus Lie Hetland. Fair allocation of conflicting items. *Auton. Agents Multi Agent Syst.*, 36(1):8, 2022.

- Ayumi Igarashi, Pasin Manurangsi, and Hirotaka Yoneda. Dividing conflicting items fairly. *arXiv preprint arXiv:2506.14149*, 2025.
- Yatharth Kumar, Sarfaraz Equbal, Rohit Gurjar, Swaprava Nath, and Rohit Vaish. Fair scheduling of indivisible chores. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pages 2345–2347, 2024.
- Bo Li, Minming Li, and Ruilong Zhang. Fair scheduling for time-dependent resources. In *NeurIPS*, pages 21744–21756, 2021.
- Lily Li, Evi Micha, Aleksandar Nikolov, and Nisarg Shah. Partitioning friends fairly. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5747–5754, 2023.
- Richard J Lipton, Evangelos Markakis, Elchanan Mossel, and Amin Saberi. On approximately fair allocations of indivisible goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, pages 125–131, 2004.
- Shengxin Liu, Xinhang Lu, Mashbat Suzuki, and Toby Walsh. Mixed fair division: A survey. *Journal of Artificial Intelligence Research*, 80:1373–1406, 2024.
- Ryoga Mahara. Existence of fair and efficient allocation of indivisible chores. *arXiv preprint arXiv:2507.09544*, 2025.
- Fangxiao Wang and Bo Li. Fair surveillance assignment problem. In *WWW*, pages 178–186. ACM, 2024.
- Yu Zhou, Tianze Wei, Minming Li, and Bo Li. A complete landscape of efx allocations on graphs: Goods, chores and mixed manna. In *33rd International Joint Conference on Artificial Intelligence (IJCAI 2024)*, pages 3049–3056. International Joint Conferences on Artificial Intelligence, 2024.

A Additional Preliminaries

A.1 Limitation of the Monotone Valuation Techniques

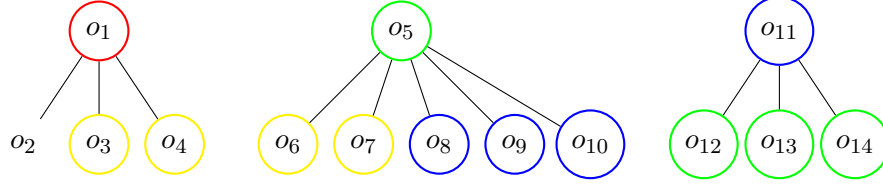
Techniques for computing EF1 allocations under monotone valuations often rely on building up the allocation incrementally, ensuring that each intermediate (partial) allocation is EF1. A key property in such approaches is that if a partial allocation is EF1, then the remaining items can be assigned in a way—possibly involving shifting bundles among agents—that preserves EF1 in the final allocation.

However, this property does not extend to non-monotone valuations, even the cut-based valuations considered in this paper. In particular, the following proposition presents a counterexample: A partial allocation that is EF1 that cannot be completed to a full allocation while maintaining the EF1 condition.

Notice that this example is based on a forest graph for which we show the existence of allocations that satisfy both EF1 and SO on forest graphs in Theorem 4.

Proposition 4. *Unlike monotone valuations, partial EF1 cannot be “completed” while maintaining EF1 for cut-valuations instances.*

Proof. Consider an instance with four agents (red, yellow, green, and blue) and 14 items where valuations are induced by the graph shown below.



A partial EF allocation is marked in the graph. Notice that object o_2 is unassigned so far. However, o_2 cannot be allocated to any of the agents without breaking EF1. This is because the red agent envies all other agents, but their value is exactly equal to that of the red agent up to the removal of the largest good. However, o_2 is a chore (negative marginal value) for agent 1 (red) and a good (positive marginal value) for all other agents. $\square$

Another common technique (under additive goods-only instances) is that of Nash welfare (the allocation maximizing the product of agents' utilities). However, our counter example for Theorem 1 also serves as an example to show that the maximum Nash welfare allocation is not EF1 for this valuation class.

A.2 Alternative EF1 definitions

The definition of EF1 that we use in this paper is used where for every pair of agent $i, j \in N$ such that $v(A_j) > v(A_i)$, there exists $o_j \in A_j$ such that $v(A_i) \geq v(A_j \setminus \{o_j\})$. This definition is typically used in goods-only instances, i.e. instances in which for every $o \in V$ and $S \subseteq V$, we have $v(S \cup \{o\}) \geq v(S)$. In settings with mixed items (i.e. goods and chores), a weaker notion is often adopted. Here, allocation A is said to be EF1 if, for every $i, j \in N$ with $v(A_j) > v(A_i)$, either there exists $o_j \in A_j$ such that $v(A_i) \geq v(A_j \setminus \{o_j\})$ or an $o_i \in A_i$ such that $v(A_i \setminus \{o_i\}) \geq v(A_j)$. That is, either there is a good in the envied agent's bundle or a chore in the envious agent's bundle whose (hypothetical) removal eliminates the envy.

It is clear that any allocation that satisfies the first definition also satisfies the other. Since we adopt the first definition, all of our positive results also give equivalent guarantees for the mixed-item variant. At the same time, we note that the counterexample for our negative results, Theorem 1, also works for the weaker (mixed-item) definition, even though a negative example for the stronger (good-based) variant does not imply a non-example for the weaker (mixed item-variant) definition.

B Non-existence of EF1 and SO for $n \geq 3$

In contrast to the non-monotonic existential guarantees for EF1 and TS allocation (always exists for $n = 2$ and $n \geq 4$ (Theorem 2), but need not exist when $n = 3$ (Theorem 1)), in this section, we provide a family of instances for every $n \geq 3$ agents such that no allocation is EF1 as well as the stronger efficiency guarantee of SO. Note that from Proposition 1, we know that EF1 and SO allocations always exist when $n = 2$.

Theorem 5 (Non-existence of EF1 and SO allocations for $n \geq 3$). *There exists an instance for every $n \geq 3$ agents for which no allocation is both EF1 and SO.*

Proof. For an instance with n agents, consider an $(n - 1)$ -partite graph constructed as follows:

- Each of the first $n - 2$ parts contains a single item which is connected to every other node in the graph.

- The final part contains $2n$ identical items. These items are not connected to each other, but each has edges to every item outside this part.

Since the graph is $(n - 1)$ -partite, any max $(n - 1)$ -cut (thus also a max n -cut) places each part in a different bundle, whereby ensuring that every edge is a cut-edge. Thus, the social welfare in any SO allocation is equal to $2|E|$.

To maximize social welfare, the $n - 2$ parts with a single item must be allocated as singletons to distinct agents, say agents $1, \dots, n - 2$. The remaining $2n$ items must then be divided among the two remaining agents. Since all the items in this part are identical, the most balanced allocation assigns n items each to agent $n - 1$ and agent n .

However, every agent $i \in [n - 2]$ EF1-envies both $n - 1$ and n , because

$$v(A_{n-1} \setminus \{o'\}) = v(A_n \setminus \{o\}) = (n - 2)(n - 1) > 2n = v(A_i),$$

for every $o \in A_n, o' \in A_{n-1}$.

Note that the inequality $(n - 2)(n - 1) > 2n$ holds for any $n \in \mathbb{Z}$, but the theorem allows for $n \geq 3$ since the construction of an $(n - 1)$ -partite graph requires there to be at least two parts. $\square$

C Missing Proofs from Section 3.3

Lemma 1. *Given an instance with identical valuations and allocation $A = (A_1, \dots, A_n)$ such that $v(A_1) \leq \dots \leq v(A_n)$, A is EF1 if and only if, for every $j \in N$ such that $v(A_j) > v(A_1)$, there exists an item $o_j \in A_j$ s.t. $v(A_1) \geq v(A_j \setminus \{o_j\})$.*

Proof. First, recall that the cut-valuations are identical for all agents. Now, to prove that the allocation A is EF1, we need to show that for any pair of agent $i, j \in N$, agent i does not envy agent j up to the (hypothetical) removal of some item $o_j \in A_j$. That is, for every pair $i, j \in N$, there exists $o_j \in A_j$ such that $v(A_i) \geq v(A_j \setminus \{o_j\})$. Since we know that $v(A_1) \leq v(A_i)$ for any i , it suffices if $v(A_1) \geq v(A_j \setminus \{o_j\})$. This is true since we know that agent 1 is not involved in any EF1 violation. In the other direction, if agent 1 is involved in an EF1 violation, then, by definition, A is not EF1. Thus, the lemma stands proved. $\square$

D Missing Proofs from Section 3.4

In this section, we present the algorithm and proof for Theorem 3.

Theorem 3 (EF1+WTS allocations). *Given a cut-valuation instance, a complete allocation satisfying EF1 and WTS always exists and can be computed in polynomial time.*

Our proof is constructive—we provide an algorithm (Algorithm 5) that computes WTS and EF1 allocations for general graphs in polynomial time. The general idea of the algorithm is as follows: Similar to Algorithm 1, we start from an arbitrary, but complete allocation. As long as the allocation is not EF1, we reallocate some items such that item(s) are transferred from the EF1-envied agent's bundles, sometimes to the least-valued bundle A_1 and sometimes to another. To satisfy WTS, items that have negative marginal values (i.e. are strict chores) in their respective bundles are transferred to some other agent. In each of these transfers, we ensure that the minimum value received by the agents ($v(A_1)$) increases by at least 1; and if it remains unchanged, the number of agents receiving the minimum value (n_1) decreases by at least 1. Thus, the vector $\Phi = (v(A_1), -n_1)$ improves lexicographically and serves as a potential function for the algorithm.

Algorithm 4 WTS-SUBROUTINE

Input: A complete allocation A s.t. $v(A_1) \leq \dots \leq v(A_n)$.

Output: A WTS allocation s.t. $v(A_1) \leq \dots \leq v(A_n)$.

- 1: **while** (there exists agent $i \in N$ with item $o \in A_i$ s.t. $v(A_i) < v(A_i \setminus \{o\})$) **do**
 - 2: If $i \neq 1$ (least valued bundle), then transfer o to A_1 ;
 - 3: Else if $i = 1$, transfer o to A_2 .
 - 4: Relabel the bundles s.t. $v(A_1) \leq \dots \leq v(A_n)$.
 - 5: **end while**
-

Since the minimum value is upper-bounded by m^2 and the number of agents is n , the algorithm returns an EF1+WTS allocation in polynomial time.

Since the valuations are identical and we can relabel the bundles, we assume, without loss of generality, that $v(A_1) \leq \dots \leq v(A_n)$ always holds at any point in the algorithm (Lemma 1). Whenever the allocation is not EF1, we first check for the following two cases (in order):

Case 1: If there is an EF1-violation towards agent i and there exists an item $o \in A_i$ that has a positive marginal value for A_1 , we reallocate o to agent 1. Note that after this reallocation, the new values of both, agent 1 and agent i , is strictly larger than the previous minimum value $v(A_1)$.

Case 2: If there are still agents whom there is an EF1-violation, but all items in their bundles have zero or non-negative marginals for A_1 . First, we note that there can be only one agent (say agent i) towards whom agent 1 has EF1-envy (Claim 2). In this case, we first find a subset $S \subset A_i$ such that $v(S)$ is strictly larger than $v(A_1)$ but there is no EF1-envy from agent 1 to S . Then, we let i keep S and reallocate all other items $A_i \setminus S$ to an agent that is not agent 1 or i (when $n \geq 3$, such an agent always exists). We show that either the new allocation is already EF1, and if not, the algorithm goes to Case 1 without decreasing Φ .

Throughout the algorithm, we repeatedly check for the weak transfer stability property WTS by checking whether any item has been allocated as a strict chore. If it is, we transfer it in a way that increases $v(A_1)$ or decreases the number of agents with value $v(A_1)$.

We note that the above algorithmic result also shows the *existence* of EF1+WTS (and hence also non-empty EF1) allocations for the class of non-monotonic valuations defined by the cut functions in a graph.

Next, we show that the WTS-SUBROUTINE terminates in polynomial time. In particular, we show that $\Phi = (v(A_1), -n_1)$ lexicographically improves.

Claim 4. *The WTS-SUBROUTINE returns an allocation that is WTS. Furthermore, in each iteration of the WTS-SUBROUTINE (Algorithm 4), the minimum value across agents ($v(A_1)$) does not decrease. Moreover, when $v(A_1)$ remains the same, the number of agents attaining the minimum value (n_1) reduces. Thus, Algorithm 4 runs in $O(m^2n)$ time.*

Proof. First, we prove that if the subroutine terminates, then the returned allocation is WTS. Recall that in a WTS allocation, there must be no transfer that strictly improves the value of both agent. Thus, we show that, in the returned allocation, the donator bundle's value either stays the same or reduces during a transfer, thereby proving that the allocation is WTS. Notice that, on termination, the while-loop condition ensures that every item is assigned to a bundle where its marginal value is non-negative. Thus, transferring any item will either reduce the value of the donator bundle or keep the value the same.

In the remaining part of the proof we argue that the while loop in the subroutine terminates in polynomial time. Towards this, we show that either the minimum value across agents ($v(A_1)$)

Algorithm 5 Computing EF1 + WTS allocations on general graphs

```
1: Initialize  $A = (A_1, \dots, A_n)$  arbitrarily with any complete allocation.
2: Relabel bundles s.t.  $v(A_1) \leq \dots \leq v(A_n)$ .
3:  $A \leftarrow \text{WTS-SUBROUTINE}(A)$ 
4: while ( $\exists$  EF1 violation from agent 1 to some other agent) do
5:   Let  $X$  be the set of agents that agent 1 EF1-envies.
6:   if ( $\exists i \in X$  and  $\exists o \in A_i$  s.t. agent 1 has positive marginal value for  $o$ ) then
7:     Transfer  $o$  to agent 1.
8:   else
9:     Initialize  $S \leftarrow \emptyset$ .
10:    while ( $v(S) \leq v(A_1)$ ) do
11:      Let  $o$  be an item in  $A_i \setminus S$  with positive marginal value for  $S$ .
12:      Update  $S \leftarrow S \cup \{o\}$ .
13:    end while
14:    Arbitrarily choose an agent  $j \neq 1, i$ .
15:    Update  $A_i \leftarrow S, A_j \leftarrow A_j \cup (A_i \setminus S)$ .
16:  end if
17:  Relabel bundles s.t.  $v(A_1) \leq \dots \leq v(A_n)$ .
18:   $A \leftarrow \text{WTS-SUBROUTINE}(A)$ .
19: end while
```

increases, or the number of agents (n_1) attaining the minimum value reduces—without decreasing the minimum value itself. The WTS-SUBROUTINE identifies a strict chore o for some agent i (an item in A_i with negative marginal value). If $i \neq 1$, then o is transferred from i 's bundle to agent 1's bundle. By Lemma 2, o must have positive marginal value for A_1 , and transferring it to A_1 increases $v(A_1)$. If there are multiple agents with the same value as $v(A_1)$, then the minimum value does not increase, but the number of agents having this value reduces. We note that $v(A_i \setminus \{o\}) > v(A_i) \geq v(A_1)$ so i 's value remains strictly above the old minimum after the transfer. Similarly, when $i = 1$, we simply remove the (strict) chore from agent 1's bundle and transfer it to any other agent (say agent 2). Again, since o is a strict chore (negative marginal value) in A_1 , it must be a good (positive marginal value) for A_j (Lemma 2), and the transfer strictly improves both agents' values. In either case, either $v(A_1)$ increases, or in the case when multiple agents attain the minimum value, the minimum value does not change but the number of agents with the minimum value decreases by at least 1. $\square$

Now we are ready to prove Theorem 3.

Proof of Theorem 3. First, we show that if Algorithm 5 terminates, then the resulting allocation is both EF1 and WTS. After that, we prove that the algorithm terminates in polynomial time. By the main while loop in Line 4, the algorithm continues until agent 1 (the agent with the least value) no longer has an EF1 violation towards any other agent. By Lemma 1, this ensures that the allocation is EF1. Next, from Claim 4, we know that the WTS-SUBROUTINE guarantees that the resulting allocation is WTS. Since this subroutine is called in Lines 3 and 18, the final allocation also satisfies WTS.

In particular, we show that (i) in every iteration of Case 1 there is a lexicographic improvement in $\Phi = (v(A_1), -n_1)$; and (ii) in Case 2, when it does not lead to a WTS allocation, calls WTS-SUBROUTINE through which Φ lexicographically improves, i.e. the allocation already WTS but not

EF1, then the algorithm moves into Case 1 without decreasing Φ . That is, there cannot be two consecutive rounds in which Case 2 is called unless Φ improves.

Since the minimum value received by the agents can increase at most $O(E) \in O(m^2)$ times, and the number of agents receiving the least value can decrease at most $O(n)$ times, the algorithm ends in $O(m^2n)$.

We first consider Case 1. Let i be an agent who is EF1-envied by agent 1 and o be an item in A_i that is a good (has positive marginal value) with respect to the set A_1 . After reallocating o to agent 1, the value that agent 1 receives strictly increases. In addition, since i was EF1-envied by agent 1, her new value $v(A_i \setminus \{o\})$ is still strictly larger than $v(A_1)$. If $v(A_2) = v(A_1)$, the minimum value does not change, but the number of agents receiving the minimum value decreases by 1. Otherwise, the minimum value increases by at least 1.

Next, we consider Case 2. Let i be the agent who is EF1-envied by agent 1, and S be a subset of A_i such that $v(S)$ is strictly larger than $v(A_1)$ but agent 1 does not EF1-envy S . Also, let $j \neq 1, i$ be the agent who receives the items in $A_i \setminus S$ along with its original bundle A_j ; one such agent always exists when $n \geq 3$. First, we show that j 's value does not decrease. This is because every item in $A_i \setminus S$ is a chore (non-positive marginal utility) for agent 1 and thus, from Lemma 2, each item in $A_i \setminus S$ has at least as many neighbors outside $A_j \cup (A_i \setminus S)$ as inside it.

Now, if the new allocation is not WTS, then Algorithm 4 is called in which, by Claim 4), improves Φ . If the new allocation after reallocating $A_i \setminus S$ to j is WTS but not EF1, by Claim 2, agent 1 only EF1-envies i before this step. Suppose that in the new allocation there is also EF1-envy towards j . From (the footnote of) Claim 2, we know that some item in A_j must be a good (positive marginal value) with respect to A_1 ; thus, the algorithm goes into Case 1. $\square$

Corollary 1 (Equitable graph cuts). *Given a graph $G = (V, E)$ and an integer $n \leq |V|$, there exists a polynomial-time computation n -partition of V into non-empty $V_1, V_2, \dots, V_n \neq \emptyset$ such that the cut-values satisfy $|v(V_i) - v(V_j)| \leq \Delta$ for all $i, j \in [n]$, where $\Delta := \max_{o \in V} \deg(o)$ denotes the maximum degree of $G = (V, E)$.*

Proof. Let $A = (A_1, \dots, A_n)$ be the outcome of Theorem 3; set $V_i \leftarrow A_i$. For any $i, j \in [n]$ such that $v(V_i) < v(V_j)$, since A is EF1, we know that

$$v(V_j \setminus \{o\}) \leq v(V_i).$$

Also, we know that the marginal benefit from any item $o \in V$ is at most the maximum degree in the graph Δ . That is,

$$v(V_j) - v(V_j \setminus \{o\}) \leq \Delta.$$

Combining both these inequalities, we get that

$$v(V_j) \leq v(V_j \setminus \{o\}) + \Delta \leq v(V_i) + \Delta.$$

That is, for any $i, j \in [n]$, we have

$$|v(V_i) - v(V_j)| \leq \Delta,$$

thereby completing the proof. $\square$