Equitable Division of a Path

Neeldhara Misra, Chinmay Sonar, P. R. Vaidyanathan and Rohit Vaish

Abstract
We study fair resource allocation under a connectedness constraint wherein a set of indivisible items are arranged on a path and only connected subsets of items may be allocated to the agents. An allocation is deemed fair if it satisfies equitability up to one good (EQ1), which requires that agents’ utilities are approximately equal. We show that achieving EQ1 in conjunction with well-studied measures of economic efficiency (such as Pareto optimality, non-wastefulness, maximum egalitarian or utilitarian welfare) is computationally hard even for binary additive valuations. On the algorithmic side, we show that by relaxing the efficiency requirement, a connected EQ1 allocation can be computed in polynomial time for any given ordering of agents, even for general monotone valuations. Interestingly, the allocation computed by our algorithm has the highest egalitarian welfare among all allocations consistent with the given ordering. On the other hand, if efficiency is required, then tractability can still be achieved for binary additive valuations with interval structure. On our way, we strengthen some of the existing results in the literature for other fairness notions such as envy-freeness up to one good (EF1), and also provide novel results for negatively-valued items or chores.

1 Introduction
The question of how to fairly divide a set of resources among agents has been extensively studied in economics, mathematics, and computer science. The formal treatment of such resource allocation problems—commonly referred to as fair division—dates back several decades [1]. There is now a rich literature on fair division problems [2, 3, 4], comprising of a variety of solution concepts and associated existential and computational results. Many of these insights have found impressive practical applications such as in rent division [5], credit assignment [6], and cluster computing [7].

Many real-world resource allocation problems exhibit a natural spatial or temporal structure, and in such scenarios, it is desirable to have contiguous allocations. For example, when allocating supercomputing time, a contiguous processing window is preferable over one that involves multiple restarts. Similarly, when assigning office space in a department building, each research group might prefer a contiguous segment of rooms for ease of communication.

In this work, we study the seemingly conflicting goals of fairness and contiguity in the context of allocating indivisible resources (or goods). Specifically, we consider a set of indivisible goods that are represented by the vertices of a path graph, and require that each agent is allocated a connected subgraph. Fair allocation of indivisible goods has received growing interest within both artificial intelligence as well as theoretical computer science literature [8, 9, 10, 11, 12], motivated, in part, by notable real-world applications such as course allocation [13] and property division [14]. The research area has been further popularized by the website Spliddit (http://www.spliddit.org/) that provides implementations of provably fair algorithms for a wide array of resource allocation problems [15].

While there are countless formulations of what it means to be fair, each with its own merit, in this work we focus on one well-established notion of fairness called equitability [16]. An equitable allocation is one in which agents derive equal utilities from their assigned shares. Equitability is a particularly compelling fairness criterion in settings such as dividing climate change responsibilities among countries [17] and in designing taxation policies. It also enjoys empirical support, as lab experiments and an online user study have found that equitability—or "aversion of interpersonal
inequity”—can be an important predictor of the perceived fairness of an allocation, possibly more so than the classic “intrapersonal” criterion of envy-freeness [18, 19]. Equitability is also a key property in the well-known adjusted winner algorithm [2] which has been applied to divorce settlements.

For indivisible items, perfect equitability may not be possible, which motivates the need for a natural relaxation called equitability up to one good (EQ1) [20]. This notion requires that the inequity between any pair of agents can be eliminated by removing some item from the happier agent’s bundle. Since an empty allocation is vacuously fair, the study of fairness notions is often coupled with economic efficiency. To this end, we study EQ1 alongside various efficiency measures such as Pareto optimality, non-wastefulness, and maximum egalitarian (max-min) or utilitarian (sum) welfare (see Preliminaries for the relevant definitions).

The study of connected fair allocations of general graphs was initiated by [21] with a focus on other fairness notions such as envy-freeness, proportionality, and maximin share. Concurrently, [22] showed that for a path graph, a connected and approximately equitable allocation always exists and can be efficiently computed. This work also provided a non-constructive proof of existence of egalitarian-optimal and approximately equitable allocations, but did not consider other efficiency notions. Importantly, the notion of approximate equitability in Suksompong’s work is strictly weaker than EQ1, and as we observe later, his algorithm could fail to find EQ1 allocations even when such allocations are known to exist. Thus, the existential and computational questions pertaining to EQ1 allocations remain unanswered by prior work.

Our Contributions: We initiate the study of EQ1 allocations under connectedness constraints and make the following contributions:

1. **Hardness results for EQ1 and efficient allocations:** We show that checking the existence of a connected EQ1 allocation satisfying any of the aforementioned efficiency measures is NP-hard even under binary additive valuations (Theorems 1 and 2 and Corollary 1). All of our results follow from a single construction that also has implications for other fairness notions such as envy-freeness up to one good (EF1) as well as negatively-valued items (or chores).

2. **Algorithmic result for complete EQ1 allocations:** By relaxing the efficiency condition and only requiring completeness (i.e., not leaving any good unassigned), we obtain a polynomial-time algorithm for computing a connected EQ1 allocation whose egalitarian welfare is the highest among all allocations that are consistent with a given ordering of agents (Theorem 3). This resolves an open problem of [22]. Notably, our algorithm applies to any instance with monotone (possibly non-additive) valuations.

3. **Structured preferences:** We provide an efficient algorithm for checking the existence of a connected, non-wasteful, and EQ1 allocation when agents have binary additive valuations with extremal interval structure (Theorem 5).

2 Related Work

Fair division problems have been classically studied in the context of divisible resources, most prominently in the cake-cutting literature; see [4, Chapter 13] for an excellent survey. There is also a vast literature on connected (or contiguous) cake-cutting, spanning various notions of fairness and economic efficiency [23, 5, 24, 25, 26, 27, 28, 29, 30]. In particular, for equitability, it is known that for any given ordering of the agents, there exists a connected equitable division of a cake consistent with the ordering [31]. Although no finite procedure can compute an exactly equitable division even without the connectedness constraint [32], it is known that an \( \varepsilon \)-equitable connected division can be computed using finite protocols [33]. Equitability has also been studied in combination with other fairness notions. For example, while there always exists a connected equitable division that
is also proportional [31], there might not exist a connected division that is simultaneously equitable and envy-free [34].

For indivisible resources, the study of connected fair division has more recent origins [35, 21, 22]. A number of fairness notions such as proportionality, envy-freeness, and maximin share have been examined in this model when the resources are goods [21, 36, 37, 38, 39, 22, 40], chores [41], and mixed items involving both goods and chores [42]. A noteworthy result in this context concerns the existence of allocations satisfying envy-freeness up to one good (EF1) when the number of agents is at most four [38], or when agents have identical valuations [38, 40]. As we observe in Remark 4, the latter result follows as a corollary of our main algorithmic result.

For indivisible goods without the connectedness requirement, [43] provided an efficient algorithm for achieving equitability up to any good. Subsequently, [20] studied (approximate) equitability along with Pareto optimality. Among other results, they showed that an EQ1 and Pareto optimal algorithm might fail to exist even with binary valuations, and provided efficient algorithms for checking the existence of such allocations. By contrast, as we show in Theorem 2, the problem becomes NP-complete when connectedness is also required.

3 Preliminaries

Let $\mathcal{N} = \{a_1, a_2, \ldots, a_n\}$ be a set of $n \in \mathbb{N}$ agents, and $G = (V, E)$ be an undirected graph. Each vertex $v \in V$ of the graph $G$ corresponds to an indivisible good (or item) with $m := |V|$ goods overall. A (connected) bundle is a set of goods $S \subseteq V$ whose corresponding vertices induce a connected subgraph of $G$. We let $\mathcal{C}(V) \subseteq 2^V$ denote the set of all connected subsets of $V$. Unless stated otherwise, we will assume that $G$ is a path given by $\{v_1, v_2, \ldots, v_m\}$ where $(v_i, v_{i+1}) \in E$ for $i \in [m-1]$.

A (connected) allocation $A : \mathcal{N} \rightarrow \mathcal{C}(V)$ assigns to each agent $a_i$ a connected bundle $A(a_i) \in \mathcal{C}(V)$ such that no good is assigned to more than one agent. We will denote an allocation as an ordered tuple $A = (A_1, A_2, \ldots, A_n)$, where $A_i := A(a_i)$. An allocation is said to be complete if it does not leave any good unassigned; that is, for any good $v$, there exists some agent $a_i$ such that $v \in A_i$. A partial allocation is one that is not complete. Unless stated explicitly otherwise, the term ‘allocation’ will refer to a complete allocation.

The preferences of agent $a_i$ are specified by a valuation function $u_i : \mathcal{C}(V) \rightarrow \mathbb{N} \cup \{0\}$. We say that the valuation functions are monotone if for any pair of connected bundles $S, S' \in \mathcal{C}(V)$ such that $S \subseteq S'$, we have $u_i(S) \leq u_i(S')$. The valuation functions are said to be additive if for each agent $a_i$ and each bundle $S \in \mathcal{C}(V)$, $u_i(S) := \sum_{v \in S} u_i(\{v\})$, where $u_i(\emptyset) := 0$. Note that since all valuations are non-negative, any additive valuation function is also monotone. We will assume throughout that the valuations are additive (however, note that our algorithmic results apply to monotone, possibly non-additive valuations). For simplicity, we will write $u_{i,j} := u_i(\{v_j\})$. An $n$-tuple of valuation functions $\mathcal{U} = \{u_1, \ldots, u_n\}$ is called a valuation profile. We say that agents have binary (additive) valuations if $u_{i,j} \in \{0, 1\}$ for all $a_i \in \mathcal{N}$ and $v_j \in V$.

Fairness notions: An allocation $A$ is said to be

- equitable (EQ) if for every pair of agents $a_i, a_k \in \mathcal{N}$, the utilities of $a_i$ and $a_k$ for their respective bundles are equal, that is, $u_i(A_i) = u_k(A_k)$,
- equitable up to one good (EQ1) if for every pair of agents $a_i, a_k \in \mathcal{N}$ such that $A_k \neq \emptyset$, there exists some good $v \in A_k$ such that $u_i(A_i) \geq u_k(A_k \setminus \{v\})$,
- envy-free (EF) if for every pair of agents $a_i, a_k \in \mathcal{N}$, $u_i(A_i) \geq u_i(A_k)$, and
The notions of EQ, EQ1, EF, and EF1 were formulated in the context of resource allocation by [16], [20], [44], and [13], respectively.\footnote{[45] studied a weaker approximation of envy-freeness than EF1, but their algorithm is known to compute an EF1 allocation.}

Notice that equitability and envy-freeness (and their corresponding relaxations) coincide when agents have \textit{identical} valuations (i.e., if \( u_i = u_k \) for every \( a_i, a_k \in \mathcal{N} \)) but are incomparable in general. Although our focus in this paper is on (approximate) equitability, some of our results also have implications for (approximate) envy-freeness.

**Efficiency notions:** An allocation \( A \) is said to be

- **Pareto optimal** (PO) if for no other connected allocation \( B \), we have \( u_i(B_i) \geq u_i(A_i) \) for every agent \( a_i \), with at least one of the inequalities being strict, and
- **non-wasteful** (NW) if for any good \( v \), there exists some agent \( a_i \) such that \( v \in A_i \) and \( u_i(\{v\}) > 0 \).\footnote{To make this notion well-defined, we will assume throughout that in any given instance, for every good there is at least one agent with a non-zero value for it. This assumption is without loss of generality as our negative results (pertaining to computational hardness and non-existence) hold even under this assumption, and our positive results (algorithms and existence results) do not need this assumption.}

The \textit{utilitarian welfare} of \( A \) is the sum of utilities of all agents in \( A \), i.e., \( \sum_{a_i \in \mathcal{N}} u_i(A_i) \), and the \textit{egalitarian welfare} of \( A \) is the utility of the least happy agent, i.e., \( \min_{a_i \in \mathcal{N}} u_i(A_i) \).

Non-wastefulness and Pareto optimality can be incomparable notions even when \( G \) is a path.\footnote{Consider three goods \( v_1, v_2, v_3 \) on a path and two agents with valuations \( u_1 = (1, 10, 0) \) and \( u_2 = (10, 1, 1) \). The allocation \( A := \{\{v_1\}, \{v_2, v_3\}\} \) is non-wasteful but is Pareto dominated by the (wasteful) allocation \( B := \{\{v_2, v_3\}, \{v_1\}\} \).}

However, for binary valuations, NW \( \Rightarrow \) PO \( \Rightarrow \) complete (since, for binary valuations, a non-wasteful allocation maximizes the utilitarian social welfare and is therefore Pareto optimal), and there are simple examples where these implications are strict.

**Connected fair division problem:** The input to this problem is a tuple \( I = (G, \mathcal{N}, \mathcal{U}) \) consisting of a graph \( G \), a set of agents \( \mathcal{N} \), and a valuation profile \( \mathcal{U} \). The goal is to determine whether \( I \) admits a connected allocation satisfying the desired notions of fairness and efficiency. Notice that if \( G \) is a clique, we recover the standard fair division model without the connectedness constraint. In this work, we will exclusively focus on the case where \( G \) is a path graph.

\((a, b)\)-sparse instances: Given any \( 1 \leq a \leq m \) and \( 1 \leq b \leq n \), we say that an instance with binary valuations is \((a, b)\)-sparse if each agent approves (i.e., has value of 1 for) at most \( a \) goods and each good is approved by at most \( b \) agents.

### 4 Hardness Results for EQ1 and Efficient Allocations

Note that in the absence of the connectedness constraint, a non-wasteful allocation can be easily computed by assigning each good to an agent that has a positive value for it. By contrast, connectedness poses a substantial computational challenge even when we are only looking to satisfy non-wastefulness (without any fairness constraints), as the problem turns out to be NP-complete (Theorem 1).

**Theorem 1.** Determining whether there exists a connected non-wasteful allocation is NP-complete for a path and a \((4, 4)\)-sparse binary valuations instance.
To prove Theorem 1, we will show a reduction from a structured version of SATISFIABILITY called LINEAR NEAR-EXACT SATISFIABILITY (LNES) which is known to be NP-complete [46]. An instance of LNES consists of $5p$ clauses (where $p \in \mathbb{N}$) denoted as follows:

$$C = \{U_1, V_1, U'_1, V'_1, \ldots, U_p, V_p, U'_p, V'_p\} \cup \{C_1, \ldots, C_p\}.$$ 

We will refer to the first $4p$ clauses as the core clauses, and the remaining clauses as the auxiliary clauses. The set of variables consists of $p$ main variables $x_1, \ldots, x_p$, and $4p$ shadow variables $y_1, \ldots, y_{4p}$.

Each core clause consists of two literals and has the following structure:

$$\forall i \in [p], U_i \cap V_i = \{x_i\} \text{ and } U'_i \cap V'_i = \{\bar{x}_i\}.$$ 

Each main variable $x_i$ occurs exactly twice as a positive literal and exactly twice as a negative literal. The main variables only occur in the core clauses. Each shadow variable makes two appearances: as a positive literal in an auxiliary clause and as a negative literal in a core clause. Each auxiliary clause consists of four literals, each corresponding to a positive occurrence of a shadow variable.

The LNES problem asks whether, given a set of clauses with the aforementioned structure, there exists an assignment $\tau$ of truth values to the variables such that exactly one literal in every core clause and exactly two literals in every auxiliary clause evaluate to TRUE under $\tau$.

**Proof.** (of Theorem 1) Let $\phi$ be an instance of LNES. We will begin with a description of the reduced instance.

**Goods:** Introduce one good for every core clause, denoted by $U_i, V_i, U'_i,$ and $V'_i$, and two goods for every auxiliary clause, denoted by $C^L_i$ and $C^R_i$. We refer to these as core and auxiliary goods, respectively. We also introduce $2p$ dummy goods $D_1, D'_1, \ldots, D_p, D'_p$ as well as $p + 1$ separator goods $S_0, S_1, \ldots, S_p$. Thus, the total number of goods is $m = 4p + 2p + 2p + (p + 1) = 9p + 1$. The goods are arranged as shown in Figure 1.

**Agents:** For every main variable $x_i$, we will introduce two agents $a_{x_i}$ and $d_{x_i}$, for the literals; these are referred to as main agents of the positive and negative type, respectively. For every $i \in [p]$, the agent $a_{x_i}$ approves (i.e., values at 1) the goods $U_i, V_i, D_i, D'_i$, while the agent $d_{x_i}$ approves the goods $U'_i, V'_i, D'_i, D_i$. We also introduce a shadow agent for every shadow variable. If $y$ is a shadow variable occurring in the core clause $U_i$ and auxiliary clause $C_j$, then the shadow agent corresponding to $y$ approves the goods $U_i, C^L_j,$ and $C^R_j$. The set of goods approved by $y$ is analogously defined if it appears in the core clauses $V_i, U'_i$, or $V'_i$. Finally, we introduce $p + 1$ separator agents $s_0, s_1, \ldots, s_p$ such that for every $i \in \{0, 1, \ldots, p\}, s_i$ only approves the separator good $S_i$. Thus, the total number of agents is $n = 2p + 4p + (p + 1) = 7p + 1$. Observe that the constructed instance is $(4, 4)$-sparse. We now turn to the proof of equivalence of the two instances.

**The Forward Direction.** Let $\tau$ be a satisfying assignment for the LNES instance. We will construct the desired allocation as follows: For every $i \in [p]$, if the main variable $x_i$ evaluates to TRUE (i.e., if $\tau(x_i) = 1$), then assign $U_i$ and $V_i$ to agent $a_{x_i}$, $D_i$ and $D'_i$ to agent $d_{x_i}$, and $U'_i$ and $V'_i$ to the (unique) shadow agents that approve these goods. Otherwise, if $\tau(x_i) = 0$, then assign $U'_i$ and $V'_i$ to agent $a_{x_i}$, $D_i$ and $D'_i$ to agent $d_{x_i}$, and $U_i$ and $V_i$ to the (unique) shadow agents that approve these goods. Additionally, for every $i \in \{0, \ldots, p\}$, assign $S_i$ to the agent $s_i$. Finally, for
every \( i \in [p] \), assign the goods \( C_i^L \) and \( C_i^R \) to the two shadow agents whose corresponding literals satisfy the auxiliary clause \( C_i \).

The above allocation assigns each good to an agent that approves it and is therefore non-wasteful. It is also easy to see that the allocation is connected: The only agents that receive more than one good under this allocation are the main agents, and they always receive either two adjacent core goods or two adjacent dummy goods.

The **Reverse Direction**. We will now show how to recover an LNES assignment given a connected and non-wasteful allocation \( A \) for the fair division instance.

Observe that due to non-wastefulness, each separator good is assigned to a unique separator agent, and the separator agents are not assigned any other goods. Thus, for every \( i \in \{0, 1, \ldots, p\} \), \( A_{s_i} = \{S_i\} \). Similarly, the \( 2p \) dummy goods \( D_1, D_1', \ldots, D_p, D_p' \) must be allocated among at least \( p \) main agents, which leaves at most \( p \) main agents for receiving the core goods. Furthermore, the separator goods prevent any shadow agent from getting more than one auxiliary good. Thus, the \( 2p \) auxiliary goods are assigned to exactly \( 2p \) shadow agents, leaving the other \( 2p \) shadow agents for receiving the core goods.

Since each core good is approved by a unique shadow agent, at most \( 2p \) core goods can be allocated among shadow agents. Thus, the remaining \( 2p \) (or more) core goods should go to the main agents. However, due to non-wastefulness, a main agent cannot get more than two core goods. Overall, this means that one set of \( p \) main agents gets exactly two core goods each (the “lucky” agents), while the other set of \( p \) main agents gets two dummy goods each (the “unlucky” agents). Notice that the two main agents corresponding to a main variable cannot both be lucky (since that would leave one or more dummy goods unassigned), nor can both be unlucky (as that would create a similar violation for the core goods).

This brings us to a natural way of deriving an assignment \( \tau \) from the allocation \( A \). If the main agent of the positive (respectively, negative) type is unlucky, then we let \( \tau(x_i) = 0 \) (respectively, \( \tau(x_i) = 1 \)). Furthermore, if \( A \) allocates a core good to a shadow agent, then the corresponding shadow variable is set to \( 0 \), while shadow variables corresponding to shadow agents who receive auxiliary goods are set to \( 1 \). Note that exactly \( 2p \) of the \( 4p \) shadow variables are set to \( 1 \). It can be verified that \( \tau \) is indeed a satisfying assignment.

Notice that the allocation obtained in the forward direction in the proof of Theorem 1 is EQ1 and EF1, and the argument for the reverse direction is driven only by non-wastefulness. Thus, we also obtain hardness results for EQ1+NW and EF1+NW allocations. Additionally, for binary additive valuations, an allocation is non-wasteful if and only if its utilitarian welfare is at least \( m \). These observations establish the hardness of a number of related problems.

**Corollary 1.** Checking the existence of a connected allocation that is (a) EQ1 and NW, (b) EF1 and NW, (c) EQ1 and has utilitarian welfare at least \( m \), or (d) EF1 and has utilitarian welfare at least \( m \) is NP-complete for a path and a \((4, 4)\)-sparse binary valuations instance.

**Remark 1.** The hardness result in Corollary 1 can be extended to multiplicative approximations of EQ1 and EF1. Given any \( \alpha \in [0, 1] \), an allocation \( A \) is said to satisfy \( \alpha\text{-EQ1} \) (respectively, \( \alpha\text{-EF1} \)) if for every pair of agents \( a_i, a_k \in N \) such that \( A_k \neq \emptyset \), there exists some good \( v \in A_k \) such that \( u_i(A_i) \geq \alpha \cdot u_k(A_k \setminus \{v\}) \) (respectively, \( u_i(A_i) \geq \alpha \cdot u_k(A_k \setminus \{v\}) \)). The reasoning is similar: The allocation in the forward direction is EQ1 as well as EF1, and hence also \( \alpha\text{-EQ1} \) and \( \alpha\text{-EF1} \). The argument in the reverse direction only uses non-wastefulness, and therefore vacuously holds for \( \alpha\text{-EQ1} \) (or \( \alpha\text{-EF1} \)). As a result, we obtain that for any rational \( \alpha \in [0, 1] \), it is NP-complete to determine the existence of a connected \( \alpha\text{-EQ1} \) (or \( \alpha\text{-EF1} \)) allocation that is non-wasteful or has utilitarian welfare at least \( m \).

\footnote{Similar approaches have been studied in the context of envy-freeness up to any good (EFX) [47, 48].}
A straightforward adaptation of the construction in Theorem 1 also gives us the following:

**Theorem 2.** Checking the existence of a connected allocation that is (a) EQ1 and PO, (b) EF1 and PO, (c) EQ1 and has egalitarian welfare at least 2, or (d) EF1 and has egalitarian welfare at least 2 is NP-complete for a path and a (6, 4)-sparse binary valuations instance.

The proof of Theorem 2 is presented in the Appendix A.

Recently, [37, Theorem 7] have shown NP-hardness of checking the existence of a connected EF1+PO allocation of a path even for binary valuations. Their construction involves items that are valued by \( \text{all} \) agents, thus requiring \( \mathcal{O}(n) \) sparsity. By contrast, our result in Theorem 2 shows hardness even for \( \mathcal{O}(1) \) sparse instances. Finally, we note that the proof of Theorem 1 can also be adapted to show NP-hardness for egalitarian or utilitarian-optimal EQ1 allocations of chores (the relevant transformation is \( u_{i,j} = u_{i,j} - 1 \)).

### 5 Algorithmic Results for Complete EQ1 Allocations

The intractability results in the previous section prompt us to relax the efficiency requirement in search of positive results, and ask the following question: Does there always exist a connected and complete EQ1 allocation of a path?

A natural approach towards this question is to start with a connected and exactly equitable division in a cake-cutting instance derived by relaxing the indivisibility constraint (such divisions are guaranteed to exist [31, 27, 50]). The fractional cake division could then be rounded to obtain a connected and approximately equitable allocation of indivisible goods. Unfortunately, there exist instances where every rounding of the fractional cake division fails to satisfy EQ1.\(^6\)

An alternative approach is to work directly with the indivisible goods instance. For a path graph, any connected allocation can be naturally associated with a left-to-right ordering of agents, say \( \sigma \). We call a connected (partial) allocation \( \sigma \)-consistent if it assigns connected bundles from left to right according to \( \sigma \). [22] has shown that there is a polynomial-time local search algorithm that, for any fixed ordering \( \sigma \) of agents, finds a connected, complete, \( \sigma \)-consistent, and approximately equitable allocation. Specifically, his algorithm computes a \( u_{\text{max}} \)-EQ allocation, where \( u_{\text{max}} \) is the highest valuation any agent has for any good, and an allocation \( A \) is \( u_{\text{max}} \)-EQ if for every \( a_i, a_k \in \mathcal{N} \), we have \( |u_i(A_i) - u_k(A_k)| \leq u_{\text{max}} \).

Notice that \( u_{\text{max}} \)-EQ is a strictly weaker guarantee than EQ1, and there exist instances where Suksompong’s algorithm fails to compute an EQ1 allocation (even though such an allocation exists).\(^7\) Thus, this approach, too, does not resolve the existence of EQ1 and complete allocations. Moreover, this algorithm could fail to satisfy standard criteria of economic efficiency. Given this limitation, [22] posed the computation of ‘approximate equitable allocations with non-trivial welfare guarantees’ as an open problem.

We address this gap by providing a polynomial-time algorithm for computing a connected, complete, and EQ1 allocation (Theorem 3). Our algorithm also provides the following economic efficiency guarantee: For any given agent ordering \( \sigma \), our algorithm returns a connected, \( \sigma \)-consistent, and EQ1 allocation whose egalitarian welfare is the highest among all connected and \( \sigma \)-consistent allocations.

---

\(^5\)For negatively-valued items (or chores), an allocation is said to satisfy EQ1 if for every pair of agents \( a_i, a_k \in \mathcal{N} \) such that \( A_i \neq \emptyset \), there exists a chore \( v \in A_i \) such that \( v_i(A_i \setminus \{v\}) \geq v_k(A_k) \) [49].

\(^6\)Consider an instance with seven goods \( v_1, \ldots, v_7 \) and three agents with identical valuations \( u = (1, 1, 1, 1, 1, 1, 1) \). Any connected and equitable division assigns \( v_1, \ldots, v_5 \) to one agent and equally divides \( v_7 \) between the other two. In any rounding, some agent will get an empty bundle, thus violating EQ1.

\(^7\)Consider the instance in Footnote 6 where \( u_{\text{max}} = 12 \). Starting with the allocation \( A := (\{\emptyset\}, \{v_1, \ldots, v_6\}, \{v_7\}) \), Suksompong’s local search algorithm immediately returns \( A \) as the output since it is \( u_{\text{max}} \)-EQ, even though it violates EQ1. Observe that the allocation \( B := (\{v_1, v_2, v_3\}, \{v_4, v_5, v_6\}, \{v_7\}) \) is EQ1 and has a higher egalitarian welfare.
allocations. In other words, a connected and egalitarian-optimal allocation for any fixed ordering of the agents is, without loss of generality, fair (i.e., EQ1) and efficiently computable.

**Theorem 3.** There is a polynomial-time algorithm for computing a connected, complete, and EQ1 allocation of a path consistent with a given ordering of agents. Furthermore, this allocation is egalitarian-optimal among all connected allocations consistent with the given ordering.

Note that the strong existence guarantee of Theorem 3 cannot be extended to EQ1 and Pareto optimal allocations. Indeed, consider an instance with five goods and three agents where 

\[ u_1 = (1, 0, 0, 0, 0), \quad u_2 = (0, 1, 0, 0, 0), \quad \text{and} \quad u_3 = (0, 0, 1, 1, 1) . \]

For \( \sigma = (1, 2, 3) \), the unique connected, \( \sigma \)-consistent, and Pareto optimal allocation is \( (\{v_1\}, \{v_2\}, \{v_3, v_4, v_5\}) \) which violates EQ1.

**ALGORITHM 1:** Algorithm for finding a connected and complete EQ1 allocation.

**Input:** An instance \( I = \langle G, N, U \rangle \) and an ordering of agents \( \sigma \).

**Output:** A connected allocation \( A \).

\[ k \leftarrow 1 \quad \triangleright \text{Start with the smallest value in the preprocessed list } L \]

\[ \theta \leftarrow u^k \quad \triangleright \text{Initialize the guess for optimal egalitarian welfare} \]

\[ \text{while } k \leq n(m + 1)^2 \text{ do} \]

\[ \text{Starting from the leftmost available good, move rightward along } G \text{ and tentatively assign a minimal connected bundle worth at least } u^{k+1} \text{ to each successive agent in } \sigma. \]

\[ \text{if the assignment in Line 4 is infeasible then} \]

\[ \text{Exit while-loop and start Phase 2.} \]

\[ \text{else} \]

\[ \theta \leftarrow u^{k+1} \quad \triangleright \text{Update the guess} \]

\[ k \leftarrow k + 1 \quad \triangleright \text{Update } k \]

\[ \triangleright \text{Phase 2: Find a } \theta \text{-unsafe agent via a left-to-right scan} \]

\[ A \leftarrow (\emptyset, \ldots, \emptyset) \]

\[ i \leftarrow 1 \]

\[ \text{while } i \leq n \text{ do} \]

\[ \text{if there exists a } \sigma \text{-consistent partial allocation that is identical to } A \text{ for the agents } a_1, \ldots, a_{i-1}, \text{ and, in addition, assigns connected bundles worth at least } u^{k+1} \text{ to } a_i \text{ and worth at least } \theta = u^k \text{ to each of } a_{i+1}, \ldots, a_n \text{ then} \]

\[ A_i \leftarrow \text{the minimal connected bundle worth at least } u^{k+1} \text{ to agent } a_i \text{ starting from the leftmost available good.} \]

\[ G \leftarrow G \setminus A_i \quad \triangleright \text{Update the set of remaining goods} \]

\[ i \leftarrow i + 1 \]

\[ \text{else} \]

\[ \text{Exit while-loop and start Phase 3.} \quad \triangleright a_i \text{ is the leftmost } \theta \text{-unsafe agent} \]

\[ \triangleright \text{Phase 3: Finalize the remaining assignments via a right-to-left scan} \]

\[ k \leftarrow n \]

\[ \text{while } k > i \text{ do} \]

\[ A_k \leftarrow \text{the minimal connected bundle worth at least } \theta = u^k \text{ to agent } a_k \text{ starting from the rightmost available good.} \]

\[ G \leftarrow G \setminus A_k \quad \triangleright \text{Update the set of remaining goods} \]

\[ k \leftarrow k - 1 \]

\[ A_i \leftarrow G \quad \triangleright \text{Assign all remaining goods to the } \theta \text{-unsafe agent } a_i \]

\[ \text{return } A \]

**Description of the algorithm:** Let \( \sigma := (a_1, a_2, \ldots, a_n) \). Our algorithm (see Algorithm 1)
Theorem 3 follows.

To compute this value, the algorithm starts with a preprocessed list of valuations (later in Remark 2, we will provide a similar analysis for general monotone valuations).

Let us start with the running time analysis of the algorithm assuming that agents have additive valuations (later in Remark 2, we will provide a similar analysis for general monotone valuations).

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure2.png}
\caption{Illustrating the notion of $\theta$-unsafe agent on an instance with binary valuations. For $\theta = u^k = 2$ and $u^{k+1} = 3$, agent $a_1$ is $\theta$-safe because there exists a partial allocation in which $a_1$’s utility is at least $u^{k+1}$ and that of each of its successors is at least $u^k$. Agent $a_2$ is $\theta$-unsafe because giving a utility of at least $u^{k+1}$ to both $a_1$ and $a_2$ necessarily involves $a_3$’s utility being less than $u^k$.}
\end{figure}

consists of three phases.

In Phase 1, the algorithm computes the optimal egalitarian welfare $\theta$ for $\sigma$-consistent allocations. To compute this value, the algorithm starts with a preprocessed list $L = \{u^1, u^2, \ldots\}$ containing all distinct realizable utility values of any agent for any connected bundle, where $u^1 := 0 < u^2 < u^3$ and so on. (The list $L$ is of length $O(nm^2)$ since the number of distinct connected bundles in a path is $O(m^2)$.) In round $k$, the algorithm checks whether there exists a connected and $\sigma$-consistent partial allocation with egalitarian welfare $u^{k+1}$. To do this, the algorithm starts from the leftmost available good and iteratively assigns minimal connected bundles to the agents $a_1, a_2, \ldots$ such that each agent gets a utility of at least $u^{k+1}$; here, minimal refers to cardinality-wise smallest bundle. If a feasible partial allocation exists, the algorithm updates its ‘guess’ of the achievable egalitarian welfare to $\theta = u^{k+1}$ and moves to round $k + 1$. Otherwise, it fixes $\theta = u^k$ and moves to Phase 2. Thus, for the instance in Figure 2, the partial allocation in round 1 ($\theta = 0$) is $\{(v_1), \{v_2, v_3\}, \{v_4\}\}$, and that in round 2 ($\theta = 1$) is $\{(v_1, v_2), \{v_3, v_4\}, \{v_5, v_6, v_7, v_8\}\}$. In round 3, the algorithm encounters infeasibility, so it fixes $\theta = 2$.

In Phase 2, the algorithm searches for a $\theta$-unsafe agent. Given any $\theta = u^k$, we say that agent $a_i$ is $\theta$-safe if there exists a connected and $\sigma$-consistent (partial) allocation in which each of $a_1, a_2, \ldots, a_i$ gets a utility of at least $u^{k+1}$, and each of $a_{i+1}, \ldots, a_n$ gets a utility of at least $u^k$. A $\theta$-unsafe agent is one that is not $\theta$-safe (see Figure 2). Note that a $\theta$-unsafe agent must exist since we know from Phase 1 that an egalitarian welfare of $u^{k+1}$ is not possible. The procedure in Phase 1 can be easily adapted to compute the leftmost $\theta$-unsafe agent, say $a_i$. Having found $a_i$, the algorithm now fixes the assignments of its predecessors $a_1, \ldots, a_{i-1}$ (but not $a_i$) by starting from the leftmost available good and iteratively assigning each agent a minimal connected bundle worth at least $u^{k+1}$. The algorithm now moves to Phase 3.

In Phase 3, the algorithm finalizes the assignments of the remaining agents via a right-to-left scan of the path $G$. Specifically, starting from the rightmost available good, the algorithm moves leftwards along $G$ and iteratively assigns minimal connected bundles worth at least $u^k$ to the agents in the reverse order $a_n, a_{n-1}$, and so on. Upon encountering $a_i$ for the second time, the algorithm assigns to it all the remaining goods, and returns the final allocation as the output. The proof of Theorem 3 follows.

Proof. (of Theorem 3) We will show that the above algorithm (Algorithm 1) satisfies the desired properties.

Let us start with the running time analysis of the algorithm assuming that agents have additive valuations (later in Remark 2, we will provide a similar analysis for general monotone valuations).
Since there are $O(m^2)$ possible connected bundles for each of the $n$ agents, the computation of $L$ requires computing the utility of $O(nm^2)$ bundles by adding up the individual utilities of the constituent goods. This amounts to a total running time of $O(nm^2 \log(u_{\max}))$ for the preprocessing phase. Further, we assume that the agents’ utilities for all possible bundles are cached so as to facilitate constant time access in the remainder of the algorithm.

The total running time for Phase 1 is $O(nm^3)$, since there are at most $n(m+1)^2$ iterations of the while-loop and each iteration involves scanning at most $m$ goods. Phase 2 involves at most $n$ iterations of the while-loop, and each iteration requires assigning connected bundles to all agents from left to right, which takes $O(m)$ time. Thus, the total running time for Phase 2 is $O(nm)$. In Phase 3, each good is considered at most once during the right-to-left scan, resulting in a running time of $O(m)$. Thus, overall, the algorithm takes $O(nm^3 + nm^2 \log(u_{\max}))$ time.

The allocation $A$ returned by the algorithm is complete because all leftover goods are allocated in the last step, and is $\sigma$-consistent because this property is maintained by the algorithm at every step. Furthermore, $A$ is also connected since the algorithm assigns connected bundles to $a_1, \ldots, a_{i-1}$ from left to right and to $a_n, \ldots, a_{i+1}$ from right to left. (The feasibility of the right-to-left assignment is guaranteed by the fact that $a_{i-1}$ is $\theta$-safe, as $a_i$ is leftmost $\theta$-unsafe agent.) Since $G$ is a path, the set of leftover goods assigned to $a_i$ in Phase 3 is also connected.

We will now argue that $A$ is egalitarian-optimal among all connected and $\sigma$-consistent allocations. First, observe that the value $\theta = u^k$ fixed at the end of Phase 1 is indeed the optimal egalitarian welfare of any connected and $\sigma$-consistent allocation. (Otherwise, by monotonicity of valuations, there must exist a connected and $\sigma$-consistent allocation with egalitarian welfare $u^{k+1}$ or higher in which agents receive minimal bundles. This, however, would contradict the infeasibility encountered for $\theta = u^{k+1}$.) Next, we will show that the egalitarian welfare of $A$ is equal to $u^k$, which will establish egalitarian-optimality. Indeed, each of $a_1, \ldots, a_{i-1}$ gets a utility of at least $u^{k+1} > u^k$ in Phase 2, and each of $a_n, \ldots, a_{i+1}$ gets a utility of at least $u^k$ in Phase 3. The utility of $a_i$ for its assigned bundle is exactly $u^k$ because of the following two reasons: First, $a_i$’s utility is at least $\theta = u_k$ since $a_{i-1}$ is $\theta$-safe (recall that $a_i$ is the leftmost $\theta$-unsafe agent). Second, since $a_i$ is $\theta$-unsafe, assigning a bundle worth at least $u^{k+1}$ to $a_i$ (and each of its predecessors $a_1, \ldots, a_{i-1}$) would imply that one of its successors $a_{i+1}, \ldots, a_n$ gets utility strictly below $u^k$, which contradicts the assignments in Phase 3. Thus, $a_i$’s utility must be strictly below $u^{k+1}$, and hence, equal to $u^k = \theta$.

Finally, to prove that $A$ is EQ1, notice that if the utility of an agent is strictly greater than $u^k$ (in particular, each of $a_1, \ldots, a_{i-1}$ gets a utility at least $u^{k+1} > u^k$), then by minimality of bundles, there must exist a boundary good whose removal results in the agent’s residual utility being strictly below $u^{k+1}$, and therefore less than or equal to $u^k$. Since each agent gets a utility at least $u^k$, $A$ must be EQ1.

We observe that the running time of Algorithm 1 can be improved to $O(nm^2)$ via following modifications: In the preprocessing step, as before, we go through all possible $O(nm^2)$ bundles. We cache these bundles, and compute $u_{\max}$ which amounts to the running time of $O(nm^2)$ for this step. Next, in Phase 1, we use binary instead of the linear search to find the optimal egalitarian welfare $\theta$. With these modifications, Phase 1 runs in time $O(m \log mn)$. In Phase 2, the $\theta$-unsafe agent can be found in $O(m)$ time with a combination of left-to-right scan that tentatively assigns bundles worth $u^{k+1}$ and a right-to-left scan that assigns bundles worth $u^k$. Finally, Phase 3 runs takes $O(m)$ time as before.

Remark 2. Note that the algorithm in Theorem 3 and the analysis of its correctness only use the monotonicity of valuations, and therefore the result extends to non-additive utilities. The running time analysis in this case relies on the existence of a valuation oracle that, given a connected bundle, returns the agent’s utility for that bundle. Since the number of distinct connected bundles
in a path is $O(m^2)$, after $O(nm^2)$ valuation queries, each agent’s value for every connected bundle is available to the algorithm. The rest of the analysis is identical to that in Theorem 3.

Remark 3. Another relevant implication is that our algorithm can be easily adapted for negative valuations to obtain the efficient computation of connected EQ1 allocations for chores. The latter result provides a tractable alternative to a recent result showing NP-hardness for connected and exactly equitable chore allocations [41].

Remark 4. [38] and [40] have independently shown that when agents have identical monotone valuations, a connected EF1 allocation of a path can be efficiently computed. Since EF1 and EQ1 coincide for identical valuations, our result in Theorem 3 implies this result as a corollary.

Additionally, we note that although the algorithm of [38] and its analysis are presented for identical valuations, a natural extension of their algorithm for general valuations can be used to derive an alternative proof of Theorem 3.

The existence result in Theorem 3 is quite general, since it applies to any fixed ordering of agents and any monotone valuations instance, and reconciles fairness (i.e., EQ1) with a weak form of economic efficiency (i.e., completeness). On closer inspection, though, we find that it implies an even stronger existence result. Specifically, given an agent ordering $\sigma$, let $A^\sigma$ denote the set of all connected, $\sigma$-consistent, EQ1 and complete allocations for the given instance. From Theorem 3, we know that $A^\sigma$ is non-empty. Furthermore, since there are only finitely many allocations, there must exist an allocation in $A^\sigma$ that is not Pareto dominated by any other allocation in $A^\sigma$. We call this property PO$^*$. In Section B, we show the following result using a variant of Algorithm 1:

Theorem 4. Given an instance with binary additive valuations and any agent ordering $\sigma$, a connected, $\sigma$-consistent, EQ1, and PO$^*$ allocation of a path can be computed in polynomial time.

6 Structured Preferences

In this section, we will explore a different avenue for circumventing the intractability associated with non-wasteful EQ1 allocations. Unlike in Theorem 3 where we relaxed the efficiency requirement, this time we will instead assume that agents have structured preferences. In particular, we will focus on binary extremal valuations wherein for each agent $a_i$, either there exists $\ell_i \in \{1, \ldots, m\}$ such that $u_{i,j} = 1$ for all $j \in \{1, \ldots, \ell_i\}$ and 0 otherwise (i.e., $a_i$ is left-extremal), or there exists $r_i \in \{m\}$ such that $u_{i,j} = 1$ for all $j \in \{r_i, \ldots, m\}$ and 0 otherwise (i.e., $a_i$ is right-extremal). Similar domain restrictions have been previously considered in the context of voting problems [51].

Theorem 5. There is a polynomial-time algorithm that, given an instance with binary extremal and additive valuations, returns a connected, non-wasteful, and EQ1 allocation whenever such an allocation exists.

Proof. We will show that the desired allocation, if it exists, can be obtained by concatenating the solutions from two subproblems, one on a purely left-extremal and the other on a purely right-extremal subsinstance.

Suppose there exists a connected, non-wasteful (NW), and EQ1 allocation $A$. Let $\sigma$ denote the agent ordering under $A$. By relabeling the agents, we have that $\sigma = (a_1, \ldots, a_n)$. We claim that without loss of generality, all left-extremal agents precede all right-extremal agents in $\sigma$. Indeed, if there is a pair of adjacent agents $a_i, a_{i+1}$ where $a_i$ is right-extremal and $a_{i+1}$ is left-extremal, then by an exchange argument we can obtain another connected, non-wasteful, and EQ1 allocation $B$ where such a violation does not occur. Specifically, by swapping the bundles of $a_i$ and $a_{i+1}$, we maintain connectedness and non-wasteful. Additionally, for binary additive valuations, non-wastefulness implies that the utility of an agent is equal to the cardinality of its bundle. Therefore,
swapping bundles results in swapping the utility values of \( a_i \) and \( a_{i+1} \), which means that the old and new allocations have identical utility profiles (up to relabeling). Thus, allocation \( B \) must also satisfy EQ1.

Let \( v_j \in V \) be such that the set \( V^L := \{v_1, \ldots, v_j\} \) is allocated among the left-extremal agents and \( V^R := \{v_{j+1}, \ldots, v_m\} \) is allocated among the right-extremal agents in \( A \). Then, the sub-instance restricted to \( V^L \) only has left-extremal valuations and admits a connected, non-wasteful, and EQ1 allocation (indeed, the restriction of \( A \) to \( V^L \) satisfies these properties). A similar implication holds for the purely right-extremal sub-instance \( V^R \). Therefore, it suffices to provide a polynomial-time algorithm for checking the existence of a connected, non-wasteful, and EQ1 allocation in a binary left-extremal instance. Notice that the same algorithm can be used for the right-extremal sub-instance via an easy ‘mirror transformation’. If both sub-instances admit desired allocations, then the concatenated allocation is clearly connected and non-wasteful in the original instance. By checking this allocation for EQ1, we obtain the desired algorithm for the original instance. Thus, in rest of the proof, we will focus only on left-extremal valuations.

Let \( A' \) denote the restriction of allocation \( A \) to the left-extremal sub-instance, and let \( n' \) and \( m' \) correspondingly denote the number of agents and items, respectively. Since \( A' \) is non-wasteful and EQ1 and the valuations are binary, the minimum and maximum utilities under \( A' \) must be \( \left\lfloor \frac{m'}{n'} \right\rfloor \) and \( \left\lceil \frac{m'}{n'} \right\rceil \), respectively. That is, an agent is either a ‘floor’ or a ‘ceiling’ agent. By an exchange argument, it can be shown that for any pair of left-extremal agents \( a_i, a_k \) such that \( i < k \), we have \( \ell_i \leq \ell_k \) (i.e., \( a_i \)'s interval finishes before \( a_k \)'s) without loss of generality. Similarly, it holds that the floor agents precede the ceiling agents (here, the exchange argument transfers a boundary item).

Let \( n'_f \) and \( n'_c \) denote the number of floor and ceiling agents, respectively. Thus, \( n'_f \) and \( n'_c \) are the unique pair of non-negative integers satisfying the equations \( n'_f + n'_c = n' \) and \( n'_f \cdot \left\lfloor \frac{m'}{n'} \right\rfloor + n'_c \cdot \left\lceil \frac{m'}{n'} \right\rceil = m' \). (If \( m' = kn' \) for some \( k \in \mathbb{N} \), then \( n'_f = n' \) and \( n'_c = 0 \).) The desired algorithm considers the agents in the order in which their intervals finish, and constructs an allocation as follows: Starting from the leftmost available good, the algorithm assigns a connected bundle of \( \left\lfloor \frac{m'}{n'} \right\rfloor \) goods to each of the first \( n'_f \) agents, and a connected bundle of \( \left\lceil \frac{m'}{n'} \right\rceil \) goods to each of the next \( n'_c \) agents. If this allocation is non-wasteful and EQ1, then the algorithm reports YES and returns the said allocation, otherwise it reports NO.

\[ \square \]

### 7 Concluding Remarks

We initiated the study of EQ1 allocations under connectedness constraints. The pursuit of connected EQ1 allocations satisfying non-trivial efficiency guarantees resulted in computational hardness. This result motivated the exploration of two avenues for tractability: relaxing the efficiency requirement and assuming structured preferences. Some of our results found broader applicability to other fairness notions (e.g., EF1) and negatively-valued items.

Going forward, it would be very interesting to explore the domain of binary intervals without the extremal structure in search of tractability results. Another relevant direction could be to map the intractability frontier for binary valuations in terms of \((a, b)\)-sparsity. Our results establish hardness of a number of problems even under \((4, 4)\)-sparsity. On the other hand, \((1, b)\)-sparse instances are efficiently solvable for any \( b \). Resolving the complexity of intermediate cases is a natural next step. Finally, extensions to general graphs [21] or settings with mixed items involving goods as well as chores [42] could also be of interest.

### References

[1] Hugo Steinhaus. The Problem of Fair Division. *Econometrica*, 16:101–104, 1948.
[2] Steven J Brams and Alan D Taylor. *Fair Division: From Cake-cutting to Dispute Resolution*. Cambridge University Press, 1996.

[3] Hervé Moulin. *Fair Division and Collective Welfare*. MIT press, 2004.

[4] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. *Handbook of Computational Social Choice*. Cambridge University Press, 2016.

[5] Francis Edward Su. *Rental Harmony: Sperner’s Lemma in Fair Division*. *The American Mathematical Monthly*, 106(10):930–942, 1999.

[6] Geoffroy De Clippel, Herve Moulin, and Nicolaus Tideman. *Impartial Division of a Dollar*. *Journal of Economic Theory*, 139(1):176–191, 2008.

[7] Ali Ghodsi, Matei Zaharia, Benjamin Hindman, Andy Konwinski, Scott Shenker, and Ion Stoica. Dominant Resource Fairness: Fair Allocation of Multiple Resource Types. In *8th USENIX Symposium on Networked Systems Design and Implementation*, volume 11, pages 24–24, 2011.

[8] Haris Aziz, Serge Gaspers, Simon Mackenzie, and Toby Walsh. Fair Assignment of Indivisible Objects under Ordinal Preferences. *Artificial Intelligence*, 227:71–92, 2015.

[9] Sylvain Bouveret and Michiel Lemaître. Characterizing Conflicts in Fair Division of Indivisible Goods using a Scale of Criteria. *Autonomous Agents and Multi-Agent Systems*, 30(2):259–290, 2016.

[10] Richard Cole and Vasilis Gkatzelis. Approximating the Nash Social Welfare with Indivisible Items. *SIAM Journal on Computing*, 47(3):1211–1236, 2018.

[11] David Kurokawa, Ariel D Procaccia, and Junxing Wang. Fair Enough: Guaranteeing Approximate Maximin Shares. *Journal of the ACM*, 65(2):1–27, 2018.

[12] 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*, 7(3):1–32, 2019.

[13] Eric Budish. The Combinatorial Assignment Problem: Approximate Competitive Equilibrium from Equal Incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.

[14] Kirk Pruhs and Gerhard J Woeginger. Divorcing Made Easy. In *International Conference on Fun with Algorithms*, pages 305–314, 2012.

[15] Jonathan Goldman and Ariel D Procaccia. Spliddit: Unleashing Fair Division Algorithms. *ACM SIGecom Exchanges*, 13(2):41–46, 2015.

[16] Lester E Dubins and Edwin H Spanier. How to Cut a Cake Fairly. *The American Mathematical Monthly*, 68(1P1):1–17, 1961.

[17] Martino Traxler. Fair Chore Division for Climate Change. *Social Theory and Practice*, 28(1):101–134, 2002.

[18] Dorothea K Herreiner and Clemens D Puppe. Envy Freeness in Experimental Fair Division Problems. *Theory and Decision*, 67(1):65–100, 2009.

[19] Ya’akov Gal, Moshe Mash, Ariel D Procaccia, and Yair Zick. Which is the Fairest (Rent Division) of Them All? *Journal of the ACM (JACM)*, 64(6):1–22, 2017.
[20] Rupert Freeman, Sujoy Sikdar, Rohit Vaish, and Lirong Xia. Equitable Allocations of Indivisible Goods. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, pages 280–286, 2019.

[21] Sylvain Bouveret, Katarína Ceclárová, Edith Elkind, Ayumi Igarashi, and Dominik Peters. Fair Division of a Graph. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, pages 135–141, 2017.

[22] Warut Suksompong. Fairly Allocating Contiguous Blocks of Indivisible Items. Discrete Applied Mathematics, 260:227–236, 2019.

[23] Walter Stromquist. How to Cut a Cake Fairly. The American Mathematical Monthly, 87(8):640–644, 1980.

[24] Xiaotie Deng, Qi Qi, and Amin Saberi. Algorithmic Solutions for Envy-Free Cake Cutting. Operations Research, 60(6):1461–1476, 2012.

[25] Xiaohui Bei, Ning Chen, Xia Hua, Biaoshuai Tao, and Endong Yang. Optimal Proportional Cake Cutting with Connected Pieces. In Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, pages 1263–1269, 2012.

[26] Yonatan Aumann, Yair Dombb, and Avinatan Hassidim. Computing Socially-Efficient Cake Divisions. In Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems, pages 343–350, 2013.

[27] Yonatan Aumann and Yair Dombb. The Efficiency of Fair Division with Connected Pieces. ACM Transactions on Economics and Computation, 3(4):23, 2015.

[28] Erel Segal-Halevi and Balázs R Sziklai. Resource-Monotonicity and Population-Monotonicity in Connected Cake-Cutting. Mathematical Social Sciences, 95:19–30, 2018.

[29] Simina Brânzei and Noam Nisan. Communication Complexity of Cake Cutting. In Proceedings of the 2019 ACM Conference on Economics and Computation, pages 525–525. ACM, 2019.

[30] Paul W Goldberg, Alexandros Hollender, and Warut Suksompong. Contiguous Cake Cutting: Hardness Results and Approximation Algorithms. Journal of Artificial Intelligence Research, 69:109–141, 2020.

[31] Katarína Ceclárová, Jozef Doboš, and Eva Pillárová. On the Existence of Equitable Cake Divisions. Information Sciences, 228:239–245, 2013.

[32] Ariel D Procaccia and Junxing Wang. A Lower Bound for Equitable Cake Cutting. In Proceedings of the 2017 ACM Conference on Economics and Computation, pages 479–495, 2017.

[33] Katarina Ceclárová and Eva Pillárová. On the Computability of Equitable Divisions. Discrete Optimization, 9(4):249–257, 2012.

[34] Steven J Brams, Michael A Jones, and Christian Klamler. Better Ways to Cut a Cake. Notices of the AMS, 53(11):1314–1321, 2006.

[35] Javier Marenco and Tomás Tetzlaff. Envy-Free Division of Discrete Cakes. Discrete Applied Mathematics, 164:527–531, 2014.

[36] Zbigniew Lonc and Mirosław Truszczynski. Maximin Share Allocations on Cycles. Journal of Artificial Intelligence Research, 69:613–655, 2020.
[37] Ayumi Igarashi and Dominik Peters. Pareto Optimal Allocation of Indivisible Goods with Connectivity Constraints. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, pages 2045–2052, 2019.

[38] 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. In 10th Innovations in Theoretical Computer Science Conference, volume 124, pages 14:1–14:21, 2019.

[39] Xiaohui Bei, Ayumi Igarashi, Xinhang Lu, and Warut Suksompong. The Price of Connectivity in Fair Division. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (forthcoming), 2021.

[40] Hoon Oh, Ariel D Procaccia, and Warut Suksompong. Fairly Allocating Many Goods with Few Queries. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, pages 2141–2148, 2019.

[41] Sylvain Bouyer, Katarína Cechlárová, and Julien Lesca. Chore Division on a Graph. Autonomic Agents and Multi-Agent Systems, 33(5):540–563, 2019.

[42] Haris Aziz, Ioannis Caragiannis, Ayumi Igarashi, and Toby Walsh. Fair Allocation of Indivisible Goods and Chores. In Proceedings of the 28th International Joint Conference on Artificial Intelligence, pages 53–59, 2019.

[43] Laurent Gourvès, Jérôme Monnot, and Lydia Tlilane. Near Fairness in Matroids. In Proceedings of the Twenty-First European Conference on Artificial Intelligence, pages 393–398, 2014.

[44] Duncan Foley. Resource Allocation and the Public Sector. Yale Economic Essays, pages 45–98, 1967.

[45] 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.

[46] Pratyush Dayal and Neeldhara Misra. Deleting to Structured Trees. In International Computing and Combinatorics Conference, pages 128–139. Springer, 2019.

[47] Benjamin Plaut and Tim Roughgarden. Almost Envy-Freeness with General Valuations. SIAM Journal on Discrete Mathematics, 34(2):1039–1068, 2020.

[48] Georgios Amanatidis, Evangelos Markakis, and Apostolos Ntokos. Multiple Birds with One Stone: Beating 1/2 for EFX and GMMS via Envy Cycle Elimination. Theoretical Computer Science, 841:94–109, 2020.

[49] Rupert Freeman, Sujoy Sikdar, Rohit Vaish, and Lirong Xia. Equitable Allocations of Indivisible Chores. In Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, pages 384–392, 2020.

[50] Guillaume Chèze. Existence of a Simple and Equitable Fair Division: A Short Proof. Mathematical Social Sciences, 87:92–93, 2017.

[51] Edith Elkind and Martin Lackner. Structure in Dichotomous Preferences. In Twenty-Fourth International Joint Conference on Artificial Intelligence, pages 2019–2025, 2015.
Neeldhara Misra  
Indian Institute of Technology Gandhinagar  
Gandhinagar, India  
Email: neeldhara.m@iitgn.ac.in

Chinmay Sonar  
University of California Santa Barbara  
Santa Barbara, CA, USA  
Email: csonar@cs.ucsb.edu

P. R. Vaidyanathan  
Technische Universität Wien, Austria  
Vienna, Austria  
Email: vaidyanathan@ac.tuwien.ac.at

Rohit Vaish  
Tata Institute of Fundamental Research  
Mumbia, India  
Email: rohit.vaish@tifr.res.in
Appendix A  Proof of Theorem 2

Let us recall the statement of Theorem 2.

**Theorem 2.** Checking the existence of a connected allocation that is (a) EQ1 and PO, (b) EF1 and PO, (c) EQ1 and has egalitarian welfare at least 2, or (d) EF1 and has egalitarian welfare at least 2 is NP-complete for a path and a (6, 4)-sparse binary valuations instance.

We will start by discussing the proof of part (a) of Theorem 2, followed by that of part (c) which uses the same construction. Parts (b) and (d) use a slightly different construction, and their proofs will be presented subsequently.

**Proof.** (of part (a)) We will show a reduction from LINEAR NEAR-EXACT SATISFIABILITY (LNES) and our construction will be similar to that of Theorem 1. Recall that an instance of LNES consists of $5p$ clauses (where $p \in \mathbb{N}$) denoted as follows:

$$\mathcal{C} = \{U_1, V_1, U'_1, V'_1, \ldots, U_p, V_p, U'_p, V'_p\} \cup \{C_1, \ldots, C_p\}.$$  

We will refer to the first $4p$ clauses as the core clauses, and the remaining clauses as the auxiliary clauses. The set of variables consists of $p$ main variables $x_1, \ldots, x_p$ and $4p$ shadow variables (our notation for the shadow variables will differ slightly from that used in Theorem 1).

Each core clause consists of two literals and has the following structure:

$$\forall i \in [p], U_i \cap V_i = \{x_i\} \text{ and } U'_i \cap V'_i = \{\bar{x}_i\}.$$  

Each main variable $x_i$ occurs exactly twice as a positive literal and exactly twice as a negative literal. The main variables only occur in the core clauses. Each shadow variable makes two appearances: as a positive literal in an auxiliary clause and as a negative literal in a core clause. For $i \in [p]$, we will let $p_i, r_i, q_i,$ and $s_i$ denote the shadow variables that appear (as negative literals) in the core clauses $U_i, V_i, U'_i$ and $V'_i$, respectively. That is, $U_i := (p_i \land x_i), V_i := (r_i \land x_i), U'_i := (q_i \land x_i),$ and $V'_i := (s_i \land x_i).$ Each auxiliary clause consists of four literals, each corresponding to a positive occurrence of a shadow variable.

The LNES problem asks whether, given a set of clauses with the aforementioned structure, there exists an assignment $\tau$ of truth values to the variables such that exactly one literal in every core clause and exactly two literals in every auxiliary clause evaluate to true under $\tau$.

**Construction of the reduced instance.** Let $\phi$ be an instance of LNES. We will begin with the description of the reduced instance.

**Goods:** For every $i \in [p]$, we introduce one good for every core clause denoted by $U_i, V_i, U'_i,$ and six goods for every auxiliary clause denoted by $C_1^{L1}, C_1^{L2}, S_i^1, S_i^2, C_i^{R1}, C_i^{R2}$. We refer to $U_i, V_i, U'_i, V'_i$ as the core goods, $C_i^{L1}, C_i^{L2}, C_i^{R1}, C_i^{R2}$ as the auxiliary goods, and $S_i^1, S_i^2$ as the separator goods. Next, we introduce two goods for each shadow variable, i.e., corresponding to each of $p_i, q_i, r_i, s_i$, we introduce the following shadow goods: $p_i^1, p_i^2, q_i^1, q_i^2, r_i^1, r_i^2, s_i^1, s_i^2$. Finally, we introduce $2p$ dummy goods denoted by $D_1, D'_1, \ldots, D_p, D'_p$, two additional separator goods $S_0^1, S_0^2$, and three special goods $S_1, S_2, S_3$. Thus, the total number of goods is $m = 4p + 6p + 8p + 2p + 2 + 3 = 20p + 5$. The goods are arranged as shown in Figure 3.

**Agents:** For every main variable $x_i$, we will introduce two agents $a_{x_i}$ and $a_{\bar{x}_i}$ for the two literals; they are referred to as main agents of the positive and negative type, respectively. For every $i \in [p]$, the agent $a_{x_i}$ approves (i.e., values at 1) the goods $U_i, V_i, D_i, D'_i$, while the agent $a_{\bar{x}_i}$ approves the goods $U'_i, V'_i, D_i, D'_i$. We also introduce a shadow agent for every shadow variable. If
The valuations of the other shadow agents
In any Lemma 1.

than the special agent achieve a utility of exactly 2

its bundle should necessarily include separator goods whose assignment has already /f_ixed.

\( S_0^1, S_0^2, C_{i1}^1, C_{i1}^2, S_1^1, S_1^2, S_{i1}^1, C_{i1}^{R1}, C_{i1}^{R2}, \ldots, C_{p1}^1, C_{p1}^2, S_p^1, S_p^2, C_{p1}^{R1}, C_{p1}^{R2} \) (Separator and auxiliary goods)

\( D_1, D_1', D_2, D_2', \ldots, D_p, D_p', S_1, S_2, S_3 \) (Dummy and special goods)

\[ p_i \] is a shadow variable occurring in core clause \( U_i \) and auxiliary clause \( C_j \), then the corresponding shadow agent \( p_i \) approves the shadow goods \( p_i^1, p_i^2 \) and the auxiliary goods \( C_{i1}^{L1}, C_{i1}^{L2}, C_{i1}^{R1}, C_{i1}^{R2} \). The valuations of the other shadow agents \( r_i, q_i, s_i \) are defined analogously. Next, we introduce \( p + 1 \) separator agents \( t_0, \ldots, t_p \) such that for every \( i \in \{0\} \cup [p] \), \( t_i \) approves two separator goods \( S_i^1, S_i^2 \). Lastly, we introduce special agent \( a_s \) that approves the special goods \( S_1, S_2, S_3 \).

This completes the construction of our reduction. Notice that the constructed instance is \((6,4)\)-sparse. Before presenting the proof of equivalence, we will establish in Lemma 1 that each agent (except for the special agent) has a utility of 2 under any EQ1 and Pareto optimal allocation.

**Lemma 1.** In any EQ1 + PO allocation, the utility of the special agent \( a_s \) is equal to 3 and that of every other agent is equal to 2.

**Proof.** (of Lemma 1) Notice that in any Pareto optimal allocation \( A \), the special goods \( S_1, S_2, S_3 \) must be allocated to the special agent \( a_s \). This is because these goods lie at the end of the path and are uniquely valued by \( a_s \), and therefore any allocation \( A' \) that does not assign these goods to \( a_s \) can be shown to be Pareto dominated by another allocation that is identical to \( A' \) except for the assignment of the special goods to the special agent. Therefore, the utility of \( a_s \) under Pareto optimal allocation must be equal to 3 (recall that \( a_s \) does not value any good other than the special goods).

Now let \( A \) denote any EQ1 and Pareto optimal allocation. Since the utility of the special agent in \( A \) is equal to 3, EQ1 requires that the utility of every other agent in \( A \) is at least 2.

Since each separator agent \( t_0, t_1, \ldots, t_p \) approves exactly two goods, it must be that for every \( i \in \{0, 1, \ldots, p\} \), the separator goods \( S_i^1, S_i^2 \) are assigned to \( t_i \) in \( A \). Furthermore, since the separator goods \( S_1^1, S_3^2 \) are placed next to each other on the path and these are the only goods approved by \( t_i \), we can assume, without loss of generality, that these are the only goods assigned to \( t_i \).

Now consider a shadow agent \( p_i \) that appears in the core clause \( U_i \) and the auxiliary clause \( C_j \). Thus, \( p_i \) approves two shadow goods \( p_i^1, p_i^2 \) and four auxiliary goods \( C_{i1}^{L1}, C_{i1}^{L2}, C_{i1}^{R1}, C_{i1}^{R2} \). Note that \( p_i \) cannot receive more than two approved goods; if it does, then by connectedness constraint, its bundle should necessarily include separator goods whose assignment has already been fixed. Thus, each shadow agent \( p_i \) (analogously \( q_i, r_i, s_i \)) will have a utility of exactly 2 in \( A \).

A similar argument shows that for any \( i \in [p] \), the main agent of positive (or negative) type \( a_x \) (or \( a_o \)) will have a utility of at most 2 since all such agents approve two core goods and two dummy goods. We therefore have that in any EQ1 and Pareto optimal allocation, all agents other than the special agent achieve a utility of exactly 2. This completes the proof of Lemma 1. \( \square \)

**The Forward Direction.** Given a satisfying assignment \( \tau \) for LNES, we will construct the desired allocation as follows:
• Allocate the special goods $S_1, S_2, S_3$ to the special agent $a_s$.

• For each $i \in \{0, 1, \ldots, p\}$, the separator agent $t_i$ receives the separator goods $S^{i}_0$ and $S^{i}_2$.

• If $\tau(\bar{x}_i) = 1$, then allocate $\{U_i, p^i_1, p^i_2, r^i_1, r^i_2, V_i\}$ to agent $x_i$ and $\{D_i, D'_i\}$ to agent $a_{x_i}$. In addition, allocate $\{U'_i, q^i_1, q^i_2\}$ to $q_i$, and $\{s^i_1, s^i_2, V'_i\}$ to $s_i$. Recall that $q_i$ and $s_i$ are the shadow variables that appear as negated literals in the core clauses $U'_i$ and $V'_i$, respectively, along with $\bar{r}_i$.

Otherwise, if $\tau(\bar{x}_i) = 0$, then allocate $\{U'_i, q^i_1, q^i_2, s^i_1, s^i_2, V'_i\}$ to agent $a_{x_i}$ and $\{D_i, D'_i\}$ to agent $a_{x_i}$. In addition, allocate $\{U_i, p^i_1, p^i_2\}$ to $p_i$, and $\{r^i_1, r^i_2, V_i\}$ to $r_i$.

• Finally, for every $j \in [p]$, allocate the sets $\{C_{j1}^{\ell_1}, C_{j2}^{\ell_2}\}$ and $\{C_j^{R_1}, C_j^{R_2}\}$ to the two shadow agents whose corresponding literals satisfy the auxiliary clause $C_j$.

Observe that each good is assigned to exactly one agent in the aforementioned allocation. Furthermore, each agent’s bundle is connected; in particular, each shadow agent either receives a set of adjacent core and shadow goods (if the corresponding shadow variable evaluates to false under $\tau$), or a set of adjacent auxiliary goods (if it evaluates to true).

It is easy to verify that the utility of the special agent is equal to 3, and that of every other agent is equal to 2. Thus, the allocation is EQ1.

We will now argue that the above allocation, say $A$, is Pareto optimal. Suppose, for contradiction, that another allocation $A'$ Pareto dominates $A$. Since the special agent and each separator agent receives all of its approved goods under $A$, the utilities of these agents under $A$ and $A'$ must be equal. Furthermore, if a main agent has a strictly higher utility under $A'$, then by the connectedness constraint, its bundle must contain a separator good, which leads to an infeasible assignment since these goods are necessarily allocated to the separator agents. A similar argument shows that a shadow agent, too, cannot receive a higher utility under $A'$. Therefore, $A$ must be Pareto optimal.

The Reverse Direction. We will now show how to recover an LNES assignment given a connected EQ1 and Pareto optimal allocation, say $A$.

Since $A$ is EQ1 and Pareto optimal, we know from Lemma 1 that the special agent receives three approved goods and every other agent receives two approved goods under $A$. Thus, in particular, for every $i \in \{0, 1, \ldots, p\}$, the separator goods $S^{i}_0, S^{i}_2$ are allocated to the separator agent $t_i$. Along with the connectedness constraint, this implies that for every $i \in [p]$, at least one of the main agents $a_{x_i}$ or $a_{x_{i+1}}$ will achieve a utility of 2 by either receiving the interval $U_i, p^i_1, p^i_2, r^i_1, r^i_2, V_i$ or $U'_i, q^i_1, q^i_2, s^i_1, s^i_2, V'_i$. This, in turn, forces at least one pair of shadow agents—either $\{p_i, r_i\}$ or $\{q_i, s_i\}$—to obtain their utilities from the auxiliary goods.

We will now show that exactly one of these two pairs of agents derive their utility from the shadow goods, while the other pair meets the utility requirement through the auxiliary goods. Indeed, since there are $4p$ auxiliary goods (corresponding to $p$ auxiliary clauses), at most $2p$ shadow agents can obtain the desired utility from the auxiliary goods. Therefore, for every $i \in [p]$, exactly one pair of shadow agents—either $\{p_i, r_i\}$ or $\{q_i, s_i\}$—are assigned shadow goods, while the other pair receives auxiliary goods. Note that this observation also shows that for every $i \in [p]$, exactly one out of $a_{x_i}$ or $a_{x_{i+1}}$ is assigned the dummy goods $\{D_i, D'_i\}$.

Overall, we have that one set of $p$ main agents gets exactly two core goods each (we will refer to them as the “lucky” agents), while the other set of $p$ main agents gets two dummy goods each (the “unlucky” agents). Notice that the two main agents corresponding to a main variable cannot both be lucky, nor can both be unlucky due to the argument presented earlier.

This brings us to a natural way of deriving an LNES assignment $\tau$ from the allocation $A$. If the main agent of the positive (respectively, negative) type is unlucky, then we let $\tau(\bar{x}_i) = 0$...
we can now infer a satisfying LNES who receive auxiliary goods are set to $\tau$ (respectively, $x$) result in Lemma 2. Instance is $S$ are defined analogously. Lastly, we introduce a shadow agent $D$ introduction of the special goods, which are all assigned to the special agent. It is easy to see that other agent, we can modify $A$ to obtain another allocation, say $A'$, that is identical to $A$ except for the allocation of the special goods, which are all assigned to the special agent. It is easy to see that $A'$ is connected, EQ1, and has egalitarian welfare at least 2. By an identical argument as in part (a), we can now infer a satisfying LNES assignment.

We now move on to the proof of part (b) of Theorem 2, followed by that of part (d) which uses a similar construction.

Proof. (of part (c)) To prove part (c), we first observe that the argument in the forward direction remains the same as in part (a), since the allocation constructed in the proof is EQ1 and satisfies the desired egalitarian welfare condition.

In the reverse direction, it is possible that under the given allocation, say $A$, the special agent $a_s$ no longer receives all three special goods. However, since the egalitarian welfare of $A$ is at least 2, each agent must receive at least two approved goods. Along with connectedness, this means that either $S_1$ or $S_3$ is not assigned to $a_s$ under $A$. Since the special goods are not approved by any other agent, we can modify $A$ to obtain another allocation, say $A'$, that is identical to $A$ except for the allocation of the special goods, which are all assigned to the special agent. It is easy to see that $A'$ is connected, EQ1, and has egalitarian welfare at least 2. By an identical argument as in part (a), we can now infer a satisfying LNES assignment.

Proof. (of part (b)) We will once again show a reduction from LINEAR NEAR-EXACT SATISFIABILITY (LNES).

**Construction of the reduced instance.** Let $\phi$ be an instance of LNES. We will begin with the description of the reduced instance.

**Goods:** For every $i \in [p]$, we introduce one core good for every core clause denoted by $U_i$, $V_i$, $U_i'$, $V_i'$, and two auxiliary goods for every auxiliary clause denoted by $C_i^L$, $C_i^R$. Next, we introduce two goods for each shadow variable, i.e., corresponding to each of $p_i$, $q_i$, $r_i$, $s_i$, we introduce the shadow goods $p_i^1, p_i^2, r_i^1, r_i^2, q_i^1, q_i^2, s_i^1, s_i^2$. Finally, we introduce $3p$ dummy goods $D_1^1, D_1^2, D_1^3, \ldots, D_p^1, D_p^2, D_p^3$ and two separator goods $S_1^0, S_2^0$. Thus, the total number of goods is $m = 4p + 2p + 8p + 3p + 2 = 17p + 2$. The goods are arranged as shown in Figure 4.

**Agents:** As before, we have the main agents of the positive and negative type for every main variable $x_i$, denoted by $a_{x_i}$ and $a_{\neg x_i}$, respectively. For every $i \in [p]$, the agent $a_{x_i}$ approves the goods $U_i, V_i, D_i^1, D_i^2, D_i^3$, while the agent $a_{\neg x_i}$ approves the goods $U_i', V_i', D_i^1, D_i^2, D_i^3$. We also introduce a shadow agent for every shadow variable. If $p_i$ is a shadow variable occurring in core clause $C_j$ and auxiliary clause $C_i$, then the corresponding shadow agent $p_i$ approves the shadow goods $p_i^1, p_i^2$ and the auxiliary goods $C_i^L, C_i^R$. The valuations of the other shadow agents $r_i, q_i, s_i$ are defined analogously. Lastly, we introduce a separator agent $a_0$ that approves the two separator goods $S_1, S_2$. This completes the construction of the reduced instance. Notice that the constructed instance is $(5,4)$-sparse. Before presenting the proof of equivalence, we will prove a structural result in Lemma 2.
Lemma 2. In any EF1 + PO allocation, the utility of the separator agent $a_0$ is equal to 2. Moreover, for every $i \in [p]$, exactly one of $a_{x_i}$ or $a_{x_i}$ is allocated the triplet of goods $\{D^1_i, D^2_i, D^3_i\}$.

Proof. (of Lemma 2) Observe that in any EF1 and Pareto optimal allocation $A$, the separator goods $S_1, S_2$ must be allocated to separator agent $a_0$. Indeed, $S_1, S_2$ are valued only by $a_0$. If $S_1, S_2$ are allocated to two distinct agents in some allocation $A'$, then $A'$ can be shown to be Pareto dominated by another allocation identical to $A'$ except for the assignment of separator goods to the separator agent. Otherwise, if $S_1, S_2$ are allocated to the same agent (different from $a_0$) in $A'$, then EF1 is violated from $a_0$’s perspective. Therefore, the utility of the separator agent $a_0$ under any EF1 and Pareto optimal allocation is equal to 2. This implies that no main or shadow agent can obtain utility from goods in both rows of Figure 4.

To prove the second part of the lemma, we first observe that for every $i \in [p]$, the goods $D^1_i, D^2_i, D^3_i$ must be assigned between the main agents $a_{x_i}$ and $a_{x_i}$ in any Pareto optimal allocation. This is because these goods are approved only by $a_{x_i}$ and $a_{x_i}$ and no other agent. Furthermore, these agents can obtain a utility of at most 2 from the core goods. Therefore, any allocation $A$ in which one or more of the dummy goods $D^1_i, D^2_i, D^3_i$ are assigned to agents other than $a_{x_i}$ and $a_{x_i}$, can be shown to be Pareto dominated by another allocation, say $A'$, that is identical to $A$ except for the assignment of these dummy goods, which are allocated exclusively among $a_{x_i}$ and $a_{x_i}$.

Next, suppose that both $a_{x_i}$ and $a_{x_i}$ are allocated only the dummy goods $D^1_i, D^2_i, D^3_i$ in a Pareto optimal allocation, say $A$. Assume, without loss of generality, that the utilities of $a_{x_i}$ and $a_{x_i}$ in $A$ are 1 and 2, respectively. Then, $A$ can be shown to be Pareto dominated by another allocation that is identical to $A$ with the exception that one of the core goods, say $U_i$, is assigned to $a_{x_i}$ and $a_{x_i}$, and the triplet $\{D^1_i, D^2_i, D^3_i\}$ to $a_{x_i}$, contradicting the Pareto optimality of $A$. Thus, the triplet of dummy goods $\{D^1_i, D^2_i, D^3_i\}$ must be completely assigned to either $a_{x_i}$ or $a_{x_i}$.

The Forward Direction. Given a satisfying assignment $\tau$ for LNES, we will construct the desired allocation as follows:

- Allocate the separator goods $S_1, S_2$ to the separator agent $a_0$.
- If $\tau(x_i) = 1$, then allocate $\{U_i, p^1_i, p^2_i, r^1_i, r^2_i, V_i\}$ to agent $a_{x_i}$ and $\{D^1_i, D^2_i, D^3_i\}$ to agent $a_{x_i}$. In addition, allocate $\{U_i, q^1_i, q^2_i\}$ to $q_i$, and $\{s^1_i, s^2_i, V_i\}$ to $s_i$. Recall that $q_i$ and $s_i$ are the shadow variables that appear as negated literals in the core clauses $U_i$ and $V_i$, respectively, along with $x_i$.
  Otherwise, if $\tau(x_i) = 0$, then allocate $\{U_i, q^1_i, q^2_i, r^1_i, r^2_i, V_i\}$ to agent $a_{x_i}$, and $\{D^1_i, D^2_i, D^3_i\}$ to agent $a_{x_i}$. In addition, allocate $\{U_i, p^1_i, p^2_i\}$ to $p_i$, and $\{r^1_i, r^2_i, V_i\}$ to $r_i$.
- Finally, for every $j \in [p]$, allocate $\{C_{j^L}\}$ and $\{C_{j^R}\}$ to the two shadow agents whose corresponding literals satisfy the auxiliary clause $C_j$.

Notice that in the constructed allocation, each good is allocated to exactly one agent, and each agent receives a connected interval. Also, the utility of the separator agent is 2, and exactly one agent corresponding to each variable receives a triplet of the corresponding dummy goods. Note that the utility of each main agent is either 2 or 3, and the utility of each shadow agent is either 1 or 2. Furthermore, any main agent is allocated at most two goods valued by any shadow agent. Hence, the constructed allocation is EF1.

We will now argue that the above allocation, say $A$, is Pareto optimal. Suppose for contradiction, that another allocation $A'$ Pareto dominates $A$. The second part of Lemma 2 implies that in any EF1 and Pareto optimal allocation, for every $i \in [p]$, the main agents $a_{x_i}$ and $a_{x_i}$ cannot both have utility 3. Thus, the utilities of the main agents under $A$ and $A'$ should be equal. Furthermore, one
of the main agents corresponding to each variable will be allocated shadow goods corresponding to a pair of shadow agents (either \(\{p_i, r_i\}\) or \(\{q_i, s_i\}\)). This implies that for at least one of these pairs, the two shadow agents should each receive a utility of 1 under \(A'\). Hence, by a similar argument as above, all shadow agents will also have the same utility under \(A\) and \(A'\), establishing that \(A'\) cannot Pareto dominate \(A\), as desired.

**The Reverse Direction.** We will now show a way to recover an LNES assignment given a connected EF1 and Pareto optimal allocation, say \(A\).

Since \(A\) is EF1 and Pareto optimal, we know from Lemma 2 that the separator agent receives the two approved goods, and for each variable \(x_i\), exactly one of the corresponding main agents \(a_{x_i}\) or \(a_{\bar{x}_i}\) receives the triplet of dummy goods \(\{D_i^1, D_i^2, D_i^3\}\). By EF1, the other main agent will achieve a utility of 2 by either receiving the interval \([U_i, p_i^1, p_i^2, r_i^1, r_i^2, V_i]\) or \([U_i, q_i^1, q_i^2, s_i^1, s_i^2, V_i']\). This, in turn, forces at least one pair of shadow agents—either \(\{p_i, r_i\}\) or \(\{q_i, s_i\}\)—to obtain their utilities from the auxiliary goods. Note that for any such pair, both agents will have a utility of at least 1 due to EF1 condition.

We will now show that exactly one of the two pairs of shadow agents derive their utility from the shadow goods, while the other pair meets the utility requirement though the auxiliary goods. Indeed, since there are \(2p\) auxiliary goods (corresponding to \(p\) auxiliary clauses), at most \(2p\) shadow agents can obtain the desired utility from the auxiliary goods. Therefore, for every \(i \in [p]\), exactly one pair of shadow agents—either \(\{p_i, r_i\}\) or \(\{q_i, s_i\}\)—are assigned shadow goods, while the other pair receives auxiliary goods.

Overall, we have that one set of \(p\) main agents gets exactly two core goods each (we will refer them as the “lucky” agents), while the other set of \(p\) main agents gets three dummy goods each (the “unlucky” agents). Notice that the two main agents corresponding to a main variable cannot both be lucky, nor can both be unlucky due to the argument presented in Lemma 2.

This brings us to a natural way of deriving an LNES assignment \(\tau\) from the allocation \(A\). If the main agent of the positive (respectively, negative) type is unlucky, then we let \(\tau(x_i) = 0\) (respectively, \(\tau(x_i) = 1\)). Furthermore, if \(A\) allocates a core good to a shadow agent, then the corresponding shadow variable is set to 0, while shadow variables corresponding to shadow agents who receive auxiliary goods are set to 1. Note that exactly \(2p\) of the \(4p\) shadow variables are set to 1 under this assignment and there are no conflicting assignments, implying that \(\tau\) is indeed a valid solution to the LNES instance. This completes the proof of part (b) of Theorem 2.

**Proof.** (of part (d)) To prove part (d), we adapt the construction in part (b) with a small change: For every \(i \in [p]\), we introduce four auxiliary goods \(C_{i,1}, C_{i,2}, C_{i,\bar{R}_1}, C_{i,\bar{R}_2}\) instead of the original two \(C_{i,1}^L, C_{i,1}^R\). Note that such an instance is \((6,4)\)-sparse. We adapt the changes in the construction to the allocation constructed in the forward direction by replacing \(C_{i,1}^L\) (respectively, \(C_{i,2}^L\)) with the set of goods \(\{C_{i,1}^{L1}, C_{i,1}^{L2}\}\) (respectively, \(\{C_{i,\bar{R}_1}, C_{i,\bar{R}_2}\}\)). In the reverse direction, it is possible that under the given allocation, say \(A\), the main agents no longer receive all three dummy goods. Similar to the argument in part (c), we can construct an allocation \(A'\) that is identical to \(A\) except that we allocate the triplet of dummy goods \(\{D_i^1, D_i^2, D_i^3\}\) to the corresponding main agent. At this stage, with a similar argument as in the reverse direction of part (b), we can recover a satisfying LNES assignment.
Appendix B EQ1 and PO* Allocations

In this section, we give a variant of Algorithm 1 to compute EQ1 and PO* allocations. Formally, given an agent ordering \( \sigma \), we say that allocation \( A \) is PO* if it is connected, \( \sigma \)-consistent, complete, and EQ1, and no other connected, \( \sigma \)-consistent, complete, and EQ1 allocation Pareto dominates \( A \). From the aforementioned argument, it follows that a PO* allocation always exists.

Intriguingly, while Algorithm 1 can be used to establish the existence of a PO* allocation even for general monotone valuations, it can fail to return such an allocation even for binary additive valuations. Indeed, consider an instance with five goods \( v_1, \ldots, v_5 \) and three agents with valuations \( u_1 = (1, 0, 0, 1, 0) \), \( u_2 = (0, 1, 1, 0, 0) \), and \( u_3 = (0, 0, 0, 1, 1) \). Given the ordering \( \sigma = (1, 2, 3) \), Algorithm 1 computes a \( \sigma \)-consistent and EQ1 allocation \( \{(v_1, v_2), (v_3, v_4), (v_5)\} \) with utility profile \((1, 1, 1)\), which is Pareto dominated by another \( \sigma \)-consistent and EQ1 allocation \( \{(v_1), (v_2, v_3), (v_4, v_5)\} \) with utility profile \((1, 2, 2)\).

Thus, for a given agent ordering \( \sigma \), PO* is stronger than EQ1-completeness as in this case, the former implies the latter. We note that PO* with respect to an ordering \( \sigma \) could be weaker (i.e., Pareto dominated) than an EQ1-complete allocation with respect to a different ordering \( \sigma' \).

This motivates the following natural question: Given an ordering \( \sigma \), can a PO* allocation be efficiently computed? While we are unable to settle this question for general monotone valuations, in Theorem 4 we show that a variant of Algorithm 1 efficiently computes a PO* allocation for binary additive valuations. Let us recall the statement of Theorem 4.

**Theorem 4.** Given an instance with binary additive valuations and any agent ordering \( \sigma \), a connected, \( \sigma \)-consistent, EQ1, and PO* allocation of a path can be computed in polynomial time.

We will start by describing the algorithm underlying this result, which, in turn, builds on Algorithm 1. This will be followed by a formal proof of Theorem 4.

**Description of the algorithm for EQ1 and PO* allocations:** Let \( \sigma := (a_1, a_2, \ldots, a_n) \). Our algorithm for Theorem 4 consists of four phases. Phases 1 and Phase 2 are identical to those in Algorithm 1, and are used to find the optimal egalitarian welfare \( \theta \) and the leftmost \( \theta \)-unsafe agent \( a_i \), respectively. Recall that in Phase 2, we also fix the allocations of the agents \( a_1, a_2, \ldots, a_{i-1} \).

In the third phase, which we denote by Phase 3*, we partition the agents \( a_1, \ldots, a_n \) in two groups as follows: We start with a partial allocation of the first \( i-1 \) agents \( a_1, a_2, \ldots, a_{i-1} \), and then consider the remaining agents sequentially from left to right. That is, in round \( j \in \{i, i+1, \ldots, n\} \), we consider the leftmost unallocated agent according to \( \sigma \), namely \( a_j \). Starting with the leftmost available good, we allocate a minimal bundle worth \( \theta + 1 \) to \( a_j \) (note that \( \theta + 1 \) is a realizable utility value under binary valuations). Next, the algorithm checks whether there exists a connected and \( \sigma \)-consistent allocation such that each subsequent agent receives utility \( \theta \) (this step is similar to that in Algorithm 1). If the check passes (i.e., if there is a feasible partial allocation where the agents \( a_{j+1}, \ldots, a_n \) receive utility \( \theta \) each), then we assign \( a_j \) to group 1 and allocate to it the minimal bundle with utility \( \theta + 1 \) (this is a temporary allocation). Otherwise, we assign \( a_j \) to group 2 and allocate to it the minimal bundle worth \( \theta \). The above procedure is repeated for all subsequent agents, following which the algorithm proceeds to the fourth phase.

In Phase 4*, we finalize the allocation of the agents \( a_1, \ldots, a_n \) (recall that the allocation in Phase 3* is only tentative). At first, we mimic the allocation for agents \( a_1, \ldots, a_{n-1} \) from Phase 3*, and allocate the remaining goods to \( a_n \). In this allocation, if \( u_n(A_n) \leq \theta + 1 \), then the algorithm finalizes the bundles of all agents and returns the allocation. Otherwise, the algorithm performs a right-to-left scan of the path \( G \) similar to Phase 3 of Algorithm 1. In particular, starting from the rightmost available good, the algorithm moves leftwards along \( G \) and iteratively assigns minimal connected
bundles with utility $\theta + 1$ to group 1 or $\theta$ to group 2 agents in the reverse order $a_n, a_{n-1}, \ldots, a_{i+1}$. The remaining goods are assigned to agent $a_i$ and the final allocation is returned as the output.

**Proof.** (of Theorem 4) First, observe that the set of goods allocated in Phase 4* at least contains all the goods allocated in a temporary allocation of Phase 3* (it may contain some additional goods). This is because, during a left-to-right partial temporary allocation in Phase 3*, we may not consider leftover goods to the right of the allocated bundle for agent $a_n$. Hence, in the final allocation in Phase 4*, algorithm has enough goods such that each group 1 agent receives a utility of at least $\theta + 1$, and each group 2 agent receives a utility of at least $\theta$.

Next, we show that the allocation $A$ returned by the algorithm is $\sigma$-consistent, complete, and EQ1. Notice that for the two cases in Phase 4*, in the last iteration, we allocate the leftover goods to agent $a_n$ or agent $a_i$; hence, completeness follows trivially. Also, $A$ is $\sigma$-consistent because the algorithm maintains this property at every step. Note that each of the agents $a_1, \ldots, a_{i-1}$ receives a bundle with utility $\theta + 1$ under allocation $A$. In Phase 4*, consider the case when final allocation is a completion of temporary partial allocation from Phase 3* by assigning leftover goods to agent $a_n$. Here, it is easy to see that the agents $a_i, \ldots, a_n$ are each allocated bundles with utility $\theta$ or $\theta + 1$. Hence, the allocation is EQ1. For the other case, when the final allocation is built with a right-to-left traversal and the leftover goods are assigned to agent $a_i$, it is easy to see that, except for agent $a_i$, all other agents receive a bundle with utility either $\theta$ or $\theta + 1$. Moreover, $a_i$ belongs to group 2 (using the definition of $\theta$-unsafe agent), and it receives a bundle with utility at least $\theta$. Now, let $S, S'$ be the set of goods allocated to agent $a_i$ under the allocation $A$, and allocation (say $A'$) by Algorithm 1 for when we run it on the same instance respectively. Observe that $S \subseteq S'$ since the agents $a_{i+1}, \ldots, a_n$ receive a minimal bundle with utility either $\theta$ or $\theta + 1$ under the allocation $A'$ while these agents receive a minimal bundle of utility exactly $\theta$ under the allocation $A$. At this stage, just like in the proof of Algorithm 1, we can conclude that $u_i(S) = \theta$. Hence, the allocation $A$ is indeed EQ1.

Finally, we show that $A$ is PO* with a proof by contradiction. Let $B$ be a $\sigma$-consistent EQ1 allocation that Pareto dominates $A$. From Theorem 3, we know that the optimal egalitarian welfare for any connected and $\sigma$-consistent allocation is $\theta$. Hence, each agent receives a bundle with utility either $\theta$ or $\theta + 1$ under allocation $A$ due to the way our algorithm works. Moreover, each agent receives a bundle with utility either $\theta$ or $\theta + 1$ under allocation $B$ as $\theta$ is the optimal egalitarian welfare for the instance and allocation $B$ is EQ1. Let $j$ be the lefmost agent such that $u_j(A_j) < u_j(B_j)$. We claim that $j > i$ where $a_i$ is the $\theta$-unsafe agent. This is because the agents $a_1, a_2, \ldots, a_{i-1}$ each receive a bundle with utility $\theta + 1$ under allocation $A$ and $a_i$ is the $\theta$-unsafe agent. It is easy to see that $u_j(B_j) = \theta + 1$. Let $A'$ be an allocation which is identical to allocation $A$ for all agents $a_1, a_2, \ldots, a_{j-1}$, and allocates a minimal connected bundle to $a_j$ such that $u_j(A'_j) = \theta + 1$. Furthermore, let $A^*$ (respectively, $B^*$) be the set of goods to the right of bundle $A'_j$ (respectively, $B_j$). We claim that $B^* \subseteq A^*$. This is because for all $\ell < j$, $u(A\ell) = u(B\ell)$, and our algorithm allocated minimal bundles. But since $u_j(A_j) = \theta$, in Phase 3*, our algorithm labeled $a_j$ as group 1 agent. This implies that the set of goods $A^*$ is not sufficient to ensure a utility $\theta$ for all subsequent agents. Since $B^* \subseteq A^*$, the allocation $B$ is not EQ1 which is a contradiction.

Finally, let us turn to the running time analysis. Since we only consider binary valuations, the list $L$ of all distinct realizable utility values contains at most $m$ distinct values, and can be precomputed in $O(nm)$ time. By a similar running time analysis as in the proof of Theorem 3, it follows that the total running time for Phase 1 is $O(m^2)$, and that for Phase 2 is $O(nm)$.

In Phase 3*, for each fixed $j$, in order to decide the group of agent $a_j$, each of the unallocated goods is considered towards at most one bundle. Hence, the total running time for this and each subsequent iteration is $O(m)$. Since there are at most $n$ iterations, the total running time for Phase 3* is $O(nm)$. In Phase 4*, we finalize the allocation of the agents $a_1, \ldots, a_n$ by constructing at
most two complete allocations corresponding to the two cases. Each good is considered towards at most one bundle in each of these allocations. Thus, the algorithm requires $O(m)$ time in Phase 4. Hence, the overall running time of our algorithm is $O(m^2 + nm)$.

We close this section by noting that the problem of computing $PO^*$ allocations remains an interesting open question for general monotone valuations. Our algorithm for this problem does not extend too far beyond the binary regime, as the following example shows: Consider an instance with four goods $v_1, v_2, v_3, v_4$ and two agents $a_1, a_2$ with valuations $u_1 = (1, 3, 1, 0), u_2 = (0, 0, 0, 2)$. Suppose the agent ordering is $\sigma = (1, 2)$. The optimal egalitarian welfare is $\theta = 2$, and $a_2$ is the left-most $\theta$-unsafe agent. On this instance, our algorithm returns the EQ1 allocation $(\{v_1, v_2\}, \{v_3, v_4\})$, which is Pareto dominated by the EQ1 allocation $(\{v_1, v_2, v_3\}, \{v_4\})$. 
