
Mathematics of Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

# Consensus Halving for Sets of Items

Paul W. Goldberg, Alexandros Hollender, Ayumi Igarashi, Pasin Manurangsi, Warut Suksompong

informs
https://pubsonline.informs.org/journal/moor

MATHEMATICS OF OPERATIONS RESEARCH
Vol. 47, No. 4, November 2022, pp. 3357–3379
ISSN 0364-765X (print), ISSN 1526-5471 (online)



# Consensus Halving for Sets of Items

**Paul W. Goldberg,[a] Alexandros Hollender,[a] Ayumi Igarashi,[b] Pasin Manurangsi,[c] Warut Suksompong[d,*]**

[a] Department of Computer Science, University of Oxford, Oxford OX1 3QD, United Kingdom; [b] Principles of Informatics Research Division, National Institute of Informatics, Tokyo 101-8430, Japan; [c] Google Research, Mountain View, California 94043; [d] School of Computing, National University of Singapore, Singapore 117417, Singapore
*Corresponding author

**Contact:** paul.goldberg@cs.ox.ac.uk, https://orcid.org/0000-0002-5436-7890 (PWG); alexandros.hollender@cs.ox.ac.uk, https://orcid.org/0000-0001-5255-9349 (AH); ayumi_igarashi@nii.ac.jp (AI); pasin@google.com (PM); warut@comp.nus.edu.sg, https://orcid.org/0000-0001-8973-2539 (WS)

**Received:** September 17, 2020
**Revised:** July 26, 2021; November 7, 2021
**Accepted:** November 19, 2021
**Published Online in Articles in Advance:** February 10, 2022

**MSC2020 Subject Classification:** Primary: 91B10, 68Q17; secondary: 68W01

https://doi.org/10.1287/moor.2021.1249



**Abstract.** Consensus halving refers to the problem of dividing a resource into two parts so that every agent values both parts equally. Prior work shows that, when the resource is represented by an interval, a consensus halving with at most $n$ cuts always exists but is hard to compute even for agents with simple valuation functions. In this paper, we study consensus halving in a natural setting in which the resource consists of a set of items without a linear ordering. For agents with linear and additively separable utilities, we present a polynomial-time algorithm that computes a consensus halving with at most $n$ cuts and show that $n$ cuts are almost surely necessary when the agents' utilities are randomly generated. On the other hand, we show that, for a simple class of monotonic utilities, the problem already becomes polynomial parity argument, directed version–hard. Furthermore, we compare and contrast consensus halving with the more general problem of consensus $k$-splitting, with which we wish to divide the resource into $k$ parts in possibly unequal ratios and provide some consequences of our results on the problem of computing small agreeable sets.



**Funding:** This work was supported by a National University of Singapore Start-up Grant, an Engineering and Physical Sciences Research Council doctoral studentship [Grant 1892947], H2020 European Research Council (ACCORD) [Grant 639945], and by the Japan Science and Technology Agency ACT-X.

**Keywords:** consensus halving • PPAD-hardness • resource allocation

## 1. Introduction

Given a set of resources, how can we divide it between two groups of agents in such a way that every member of both groups believes that the two resulting parts have the same value? This is an important problem in resource allocation and has been addressed several times under different names (Alon [1], Hobby and Rice [29], Neyman [37]) with *consensus halving* being the name by which it is best known today (Simmons and Su [50]). In the context of operations research, the two groups could be trying to agree upon a scheduling of machine processing time (Moulin [36]), a division of real estate (Segal-Halevi et al. [49]), or an allocation of other divisible resources, sometimes referred to metaphorically as a "cake" (Deng et al. [17]). Consensus halving can be seen as a strong form of group fairness, with which we require all agents to view the two parts of the resource as being equal.

In prior studies of consensus halving, the resource is represented by an interval, and the goal is to find an equal division into two parts that makes a small number of cuts in the interval.[1] Using the Borsuk–Ulam theorem from topology, Simmons and Su [50] establish that, for any continuous preferences of the $n$ agents involved, there is always a consensus halving that uses no more than $n$ cuts—this matches the number of cuts required in the worst case. In addition, the same authors develop an algorithm that computes an $\varepsilon$-approximate solution for any given $\varepsilon > 0$, meaning that the values of the two parts differ by at most $\varepsilon$ for every agent. Although the algorithm is more efficient than a brute-force approach, its running time is exponential in the parameters of the problem. This is, in fact, not a coincidence: Filos-Ratsikas and Goldberg [19] recently showed that $\varepsilon$-approximate consensus halving is polynomial parity argument (PPA)-complete, implying that the problem is unlikely to admit a polynomial-time algorithm. Filos-Ratsikas et al. [22] strengthens this result by proving that the problem remains hard even when the agents have simple valuations over the interval. In particular, the PPA-completeness result holds for agents with "two-block uniform" valuations, that is, valuation functions that are piecewise uniform over the interval and assign nonzero value to at most two separate pieces.

Whereas these hardness results stand in contrast to the positive existence result, they rely crucially on the resource being in the form of an interval. Most practical division problems do not fall under this assumption, including when we divide assets, such as houses, cars, stocks, business ownership, or facility usage. When each item is homogeneous, a consensus halving can be easily obtained by splitting every item in half. However, because splitting individual assets typically involves an overhead, for example, in managing a joint business or sharing the use of a house, we want to achieve a consensus halving while splitting only a small number of assets. Fortunately, a consensus halving that splits at most $n$ items is guaranteed to exist regardless of the number of items; this can be seen by arranging the items on a line in arbitrary order and applying the aforementioned existence theorem of Simmons and Su [50]. The bound $n$ is also tight: if each agent only values a single item and the $n$ valued items are distinct, all of them clearly need to be split. Nevertheless, given that the items do not inherently lie on a line, the hardness results from previous work do not carry over. Could it be that computing a consensus halving efficiently is possible when the resource consists of a set of items?

The complexity classes polynomial parity argument, directed version (PPAD) and PPA[2] turn out to be most important in the study of computational problems arising in economic theory. PPAD contains many diverse problems for which solutions are guaranteed to exist, but for which algorithms have high worst-case complexity. PPAD-hardness, as evidence of worst-case computational intractability, takes a central role in algorithmic game theory, starting with the works of Daskalakis et al. [14] and Chen et al. [13] in the context of Nash equilibrium computation. In the context of distributing divisible items among a set of agents, Garg et al. [25] and Chaudhury et al. [12] show PPAD-completeness for computing market-clearing prices, indicating that, whereas a polynomial-time algorithm is unlikely, containment in PPAD admits algorithms that are practically efficient via the "path-following" approach that exploits the structure of this complexity class. Deng et al. [17] give a related result in the context of fair division, showing PPAD-completeness for envy-free division of an interval among a set of agents having diverse valuations for divisions of the interval. The present paper is also concerned with the related class PPA; as a superclass of PPAD, PPA-hardness also indicates worst-case intractability of a problem. Recently, PPA has been found to characterize the complexity of problems in consensus division; for details, see Section 1.2.

### 1.1. Overview of Results

We assume throughout the paper that the resource is composed of $m$ items. Each item is homogeneous, so the utility of an agent for a (possibly fractional) set of items depends only on the fractions of the $m$ items in that set. For the most part, we consider utility functions that are additively separable across items, meaning that an agent's utility for a bundle of fractional items is the sum of utilities for each fractional item. For this overview, we focus on the more interesting case in which $n \le m$, but all of our results can be extended to arbitrary $n$ and $m$.

We begin in Section 2 by considering agents with *linear* utilities, in which the utility of each agent is additively separable across items and linear in the fraction of each item. Under this assumption, we present a polynomial-time algorithm that computes a consensus halving with at most $n$ cuts by finding a vertex of the polytope defined by the relevant constraints (Theorem 1). This positive result stands in stark contrast with our PPA-hardness when the items lie on a line (Theorem 2), which we obtain by discretizing an analogous hardness result of Filos-Ratsikas et al. [22]. We then show that improving the number of cuts beyond $n$ is difficult: even computing a consensus halving that uses at most $n-1$ cuts more than the minimum possible for a given instance is NP-hard (Theorem 3).

Next, in Section 3, we address the broader class of *monotonic* utilities, wherein an agent's utility for a set does not decrease when any fraction of an item is added to the set. For such utilities, we show that the problem of computing a consensus halving with at most $n$ cuts becomes PPAD-hard (Theorem 4), thereby providing strong evidence of its computational hardness. Perhaps surprisingly, this hardness result holds even for the class of utility functions that we call "symmetric-threshold utilities," which are very close to being linear. Such utility functions are additively separable across items, and for each item, having a sufficiently small fraction of the item is the same as not having the item at all; having a sufficiently large fraction of it is the same as having the whole item; and the utility increases linearly in between. We obtain this hardness by reducing from the "generalized circuit problem," which is used to establish the hardness of computing Nash equilibria in various settings (Chen et al. [13], Daskalakis et al. [14], Rubinstein [42]). In fact, in Appendix B, we prove that even a simplified version of the generalized circuit problem remains PPAD-hard, which, in addition to being of independent interest,[3] also significantly simplifies our reduction to the consensus halving problem. On the other hand, we present a number of positive results for monotonic utilities when the number of agents is constant in Appendix A.

In Section 4, we provide some implications of our results on the "agreeable sets" problem studied by Manurangsi and Suksompong [33]. A set of discrete items is said to be *agreeable* to an agent if the agent likes it at least as much as the complement set. Manurangsi and Suksompong [33] prove that a set of size at most $\lfloor\frac{m+n}{2}\rfloor$ that is agreeable to all agents always exists, and this bound is tight. They then give polynomial-time algorithms that compute an agreeable set matching the tight bound for two and three agents. We significantly generalize this result by exhibiting efficient algorithms for any number of agents with additive utilities across items as well as any *constant* number of agents with monotonic utilities (Theorem 6). In addition, we present a short alternative proof for the bound $\lfloor\frac{m+n}{2}\rfloor$ via consensus halving.

In Section 5, we study the more general problem of *consensus k-splitting* for agents with linear utilities. Our aim in this problem is to split the items into $k$ parts so that all agents agree that the parts are split according to some given ratios $\alpha_1, \ldots, \alpha_k$; consensus halving corresponds to the special case in which $k = 2$ and $\alpha_1 = \alpha_2 = 1/2$. Unlike for consensus halving, however, in consensus $k$-splitting, we may want to cut the same item more than once when $k > 2$, so we cannot assume without loss of generality that the number of cuts is equal to the number of items cut. For any $k$ and any ratios $\alpha_1, \ldots, \alpha_k$, we show that there exists an instance in which cutting $(k-1)n$ items is necessary (Theorem 7). On the other hand, a generalization of our consensus halving algorithm from Section 2 computes a consensus $k$-splitting with at most $(k-1)n$ cuts in polynomial time (Theorem 8), thereby implying that the bound $(k-1)n$ is tight for both benchmarks. We also illustrate a further difference between consensus $k$-splitting and consensus halving with respect to item ordering (Theorem 9).

Finally, in Section 6, we examine consensus halving and consensus $k$-splitting from a probabilistic viewpoint. Whereas our result from Section 2 reveals the difficulty of improving the number of cuts beyond $n$ in consensus halving, we establish that instances admitting a solution with fewer than $n$ cuts are rare. In particular, if the agents' utilities for items are drawn independently from nonatomic distributions (i.e., distributions that do not put a positive probability on any single point), it is almost surely the case that every consensus halving requires no fewer than $n$ cuts (Theorem 10). On the other hand, we show that, for consensus $k$-splitting, the required number of cuts is not usually $(k-1)n$ as one might expect: even when $n = 1$ and the utilities are drawn from the uniform distribution on $[0, 1]$, this number is 1 rather than $k-1$ (Theorem 11).

### 1.2. Related Work

Consensus halving falls under the broad area of *fair division*, which studies how to allocate resources among interested agents in a fair manner (Brams and Taylor [9, 10], Moulin [35]). Common fairness notions include *envy-freeness*—no agent envies another agent in view of the bundles the agents receive—and *equitability*—all agents have the same utility for their own bundle. The fair division literature typically assumes that each recipient of a bundle is either a single agent or a group of agents represented by a single preference. However, a number of recent papers consider an extension of the traditional setting to groups, thereby allowing us to capture the differing preferences within the same group as in our introductory example with two groups (Kyropoulou et al. [31], Manurangsi and Suksompong [32, 34], Segal-Halevi and Nitzan [46], Segal-Halevi and Suksompong [47, 48], Suksompong [53]). Note that a consensus halving is envy-free for all members of the two groups; moreover, it is equitable provided that the utilities of the agents are linear and normalized so that every agent has the same value for the entire set of items.

A classical fair-division algorithm that dates back over two decades is the *adjusted winner procedure*, which computes an envy-free and equitable division between two agents (Brams and Taylor [9]).[4] The procedure is suggested for resolving divorce settlements and international border disputes with one of its advantages being the fact that it always splits at most one item. Sandomirskiy and Segal-Halevi [43] investigate the problem of attaining fairness while minimizing the number of shared items and give algorithms and hardness results for several variants of the problem. As in our work, both the adjusted winner procedure and the work of Sandomirskiy and Segal-Halevi [43] assume that items are homogeneous and, as in Section 2, that the agents' utilities are linear in the fraction of each item and additively separable across items. Moreover, both of them require the assumption that all items can be shared; if some items are indivisible, then an envy-free or equitable allocation cannot necessarily be obtained.[5]

Besides consensus halving, another problem that also involves dividing items into equal parts is *necklace splitting*, which can be seen as a discrete analog of consensus halving (Alon [1], Alon and West [3], Goldberg and West [26]). In a basic version of necklace splitting, there is a necklace with beads of $n$ colors with each color having an even number of beads. Our task is to split the necklace using at most $n$ cuts and arrange the resulting pieces into two parts so that the beads of each color are evenly distributed between both parts. Observe that the difficulty of this problem lies in the spatial ordering of the beads—the problem would be trivial if the beads were unordered items as in our setting. Whereas consensus halving and necklace splitting have long been studied by mathematicians, they recently gained significant interest among computer scientists thanks in large part to new computational complexity results (Alon and Graur [2], Batziou et al. [6], Deligkas et al. [15, 16], Filos-Ratsikas and Goldberg [19, 20], Filos-Ratsikas et al. [21, 22, 23]). In particular, the PPA-completeness result of Filos-Ratsikas and Goldberg [19] for approximate consensus halving was the first such result for a problem that is "natural" in the sense that its description does not involve a general Boolean circuit.

## 2. Linear Utilities

We first formally define the problem of consensus halving for a set of items. There is a set $N = [n]$ of $n$ agents and a set $M = [m]$ of $m$ items, where $[r] := \{1, 2, \ldots, r\}$ for any positive integer $r$. A *fractional set of items* contains a fraction $x_j \in [0,1]$ of each item $j$. We are mostly interested in fractional sets of items in which only a small number of items are fractional—that is, most items have $x_j = 0$ or 1. Agent $i$ has a utility function $u_i$ that describes the agent's nonnegative utility for any fractional set of items; for an item $j \in M$, we sometimes write $u_i(j)$ to denote $u_i(\{j\})$. A *partition* of $M$ into fractional sets of items $M_1, \ldots, M_k$ has the property that, for every item $j \in M$, the fractions of item $j$ in the $k$ fractional sets sum up to exactly one.

**Definition 1.** A consensus halving is a partition of $M$ into two fractional sets of items $M_1$ and $M_2$ such that $u_i(M_1) = u_i(M_2)$ for all $i \in N$. An item is said to be *cut* if there is a positive fraction of it in both parts of the partition.

A utility function is said to be *additively separable* if the utility for any fractional set of items is the sum of the utilities for each fractional item in the set. In this section, we assume that, in addition to being additively separable, the agents' utility functions are linear. This means that, for a set $M'$ containing a fraction $x_j$ of item $j$, the utility of agent $i$ is given by $u_i(M') = \sum_{j \in M} x_j \cdot u_i(j)$. Observe that, under linearity, $M'$ forms one part of a consensus halving exactly when

$$\sum_{j \in M} x_j \cdot u_i(j) = \frac{1}{2} \sum_{j \in M} u_i(j) \qquad \forall i \in N, \tag{1}$$

where $x_j$ denotes the fraction of item $j$ contained in $M'$. As we mention in the introduction, a consensus halving with no more than $n$ cuts is guaranteed to exist regardless of the number of items. Our first result shows that such a division can be found efficiently for linear utilities.

**Theorem 1.** *For $n$ agents with linear utilities, there exists a polynomial-time algorithm that computes a consensus halving with at most* $\min\{n, m\}$ *cuts.*

**Proof.** If $n \geq m$, a partition that divides every item in half is clearly a consensus halving and makes $m = \min\{n, m\}$ cuts. We, therefore, assume from now on that $n \leq m$ and describe a polynomial-time algorithm that computes a consensus halving using no more than $n$ cuts.

The main idea of our algorithm is to start with the trivial consensus halving by which $x_1 = x_2 = \cdots = x_m = 1/2$ and then gradually reduce the number of cuts. We stop when the process cannot be continued, at which point we show that the consensus halving must contain at most $n$ cuts. Our algorithm is presented as follows.

1. Let $x_1 = x_2 = \cdots = x_m = 1/2$.
2. Let $S$ denote the set of $n$ equations $\sum_{j \in M} (y_j - \frac{1}{2}) \cdot u_i(j) = 0$ for $i \in N$ and let $T = \emptyset$.
3. While there exists a solution $(y_1, \ldots, y_m) \neq (x_1, \ldots, x_m)$ to $S \cup T$, do the following:
   a. For every $j \in M$ such that $y_j \neq x_j$, compute
   $$\gamma_j := \begin{cases} \dfrac{1 - x_j}{y_j - x_j} & \text{if } y_j > x_j; \\ \dfrac{x_j}{x_j - y_j} & \text{if } y_j < x_j. \end{cases}$$
   b. Let $j^* = \arg\min_{j \in M, y_j \neq x_j} \gamma_j$.
   c. For every $j \in M$, let $s_j := (1 - \gamma_{j^*}) \cdot x_j + \gamma_{j^*} \cdot y_j$, and update the value of $x_j$ to $s_j$.
   d. Add the equation $y_{j^*} = x_{j^*}$ to $T$.
4. Output $(x_1, \ldots, x_m)$.

Finding a solution $(y_1, \dots, y_m)$ to $S \cup T$ that is not equal to $(x_1, \dots, x_m)$ or determining that such a solution does not exist (step 3) can be done in polynomial time via Gaussian elimination.[6] Moreover, it is obvious that the other steps of the algorithm run in polynomial time.

We next prove the correctness of our algorithm, starting with arguing that $(x_1, \dots, x_m)$ forms a consensus halving. Because we start with a consensus halving $x_1 = \cdots = x_m = 1/2$, it suffices to show that each execution of the loop in step 3 preserves the validity of the solution. Observe that, because both $(x_1, \dots, x_m)$ and $(y_1, \dots, y_m)$ are solutions to Equations (1), their convex combination (in step 3c) also satisfies Equations (1). Furthermore, for each $j$ such that $y_j \neq x_j$, the value $\gamma_j$ is chosen so that, if we replace $\gamma_{j^*}$ by $\gamma_j$ in the formula for $s_j$, we would have $s_j = 1$ for the case $y_j > x_j$ and $s_j = 0$ for the case $y_j < x_j$. Because $\gamma_{j^*} \leq \gamma_j$, we have that $s_j \in [0, 1]$ for all $j$ such that $y_j \neq x_j$. In addition, the value of $x_j$ does not change for $j$ such that $y_j = x_j$. Thus, $(x_1, \dots, x_m)$ remains a consensus halving throughout the algorithm.

Finally, we are left to show that at most $n$ items are cut in the output $(x_1, \dots, x_m)$. As noted, our definition of $\gamma_j$ ensures that $x_{j^*} \in \{0, 1\}$ after the execution of step 3c. Furthermore, as the constraint $y_{j^*} = x_{j^*}$ is then immediately added to $T$, the value of $x_{j^*}$ does not change for the rest of the algorithm. As a result, every item $j$ for which the equation $y_j = x_j$ belongs to $T$ is uncut. Thus, it suffices to show that $|T| \geq m - n$ at the end of the execution.

When the while loop in step 3 terminates, $(x_1, \dots, x_m)$ must be the unique solution to $S \cup T$. Recall that a system of linear equations with $m$ variables can only have a unique solution when the number of constraints is at least $m$. This means that $|S \cup T| \geq m$ at the end of the algorithm. Because $|S| = n$, we must have $|T| \geq m - n$, as desired. □

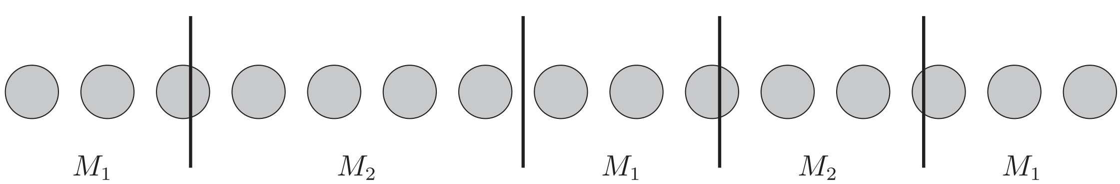
Note that this algorithm can be viewed as finding a vertex of the polytope defined by Constraints (1) and $0 \leq x_j \leq 1$ for all $j \in M$. In fact, it suffices to use a generic algorithm for this task; however, to the best of our knowledge, such algorithms often involve solving a linear program, whereas the algorithm presented here is conceptually simple and can be implemented directly. We also remark that our algorithm works even when some utilities $u_i(j)$ are negative, that is, some of the items are goods, whereas others are chores. Allocating a combination of goods and chores has received increasing attention in the fair division community (Aziz et al. [5], Bogomolnaia et al. [8], Segal-Halevi [45]).

As we discuss in the introduction, an important reason behind the positive result in Theorem 1 is the lack of linear order among the items. Indeed, as we show next, if the items lie on a line and we are only allowed to cut the line using $n$ cuts, finding a consensus halving becomes computationally hard. This follows from discretizing the hardness result of Filos-Ratsikas et al. [22] and holds even if we allow the consensus halving to be approximate instead of exact. Formally, when the items lie on a line, we may place a number of cuts with each cut lying either between two adjacent items or at some position within an item. All (fractional or whole) items between any two adjacent cuts must belong to the same fractional set of items in a partition, in which the left and right ends of the line also serve as cuts in this requirement (see Figure 1 for an example). We say that a partition into fractional sets of items $(M_1, M_2)$ is an *ε-approximate* consensus halving if $|u_i(M_1) - u_i(M_2)| \leq \varepsilon \cdot u_i(M)$ for every agent $i$.

**Theorem 2.** *Suppose that the items lie on a line. There exists a polynomial $p$ such that finding a $1/p(n)$-approximate consensus halving for $n$ agents with at most $n$ cuts on the line is PPA-hard even if the valuations are binary and every agent values at most two contiguous blocks of items.*

**Proof.** We prove this by discretizing the hard instances constructed by Filos-Ratsikas et al. [22, theorem 2]. In their setting, there are $n$ agents who have piecewise-uniform valuation functions $v_1, \dots, v_n$ over the interval $[0, 1]$.[7] By a closer inspection of their proof, we note that the instances they construct have some useful properties. Namely, there exist polynomials $p$ and $q$ such that

**Figure 1.** Consensus halving for items on a line: in this example, there are 15 items (represented by gray balls) that lie on a line, and we have used four cuts to obtain a partition into fractional sets of items ($M_1$, $M_2$). The labels $M_1$ and $M_2$ indicate the set to which each segment belongs.



1. Every agent has a two-block uniform valuation on $[0,1]$; that is, the density of the valuation function is piecewise-uniform and nonzero in at most two intervals. In other words, every agent has (at most) two blocks of value, and they have the same height.

2. There exists an integer $d \le q(n)$ such that, for all agents, the endpoints of the blocks are rational numbers with denominator $d$.

3. Finding a $1/p(n)$-approximate consensus halving is PPA-hard.

Using these properties, we can construct an equivalent instance in our setting. We position $m = d$ items on a line, where the $j$th item represents the interval $I_j := [(j-1)/d, j/d]$ in the original instance. Note that, for every agent of the original instance, the density of their valuation function is constant over $I_j$ for each $j$. Thus, by letting

$$u_i(j) = \begin{cases} 1 & \text{if } v_i((j-1)/d, j/d) > 0; \\ 0 & \text{if } v_i((j-1)/d, j/d) = 0 \end{cases}$$

for all $i \in [n]$ and $j \in [d]$, we have exactly recreated the same valuation functions in our setting, up to normalization. In particular, any $1/p(n)$-approximate consensus halving of the items using at most $n$ cuts on the line immediately yields a $1/p(n)$-approximate consensus halving of $v_1, \dots, v_n$ using at most $n$ cuts on $[0,1]$, implying that our problem is also PPA-hard. □

Although Theorem 1 allows us to efficiently compute a consensus halving with no more than $n$ cuts in any instance, for some instances, there exists a solution using fewer cuts. An extreme example is when all agents have the same utility function, in which case a single cut already suffices. This raises the question of determining the least number of cuts required for a given instance. Unfortunately, when there is a single agent, deciding whether there is a consensus halving that leaves all items uncut is already equivalent to the well-known NP-hard problem PARTITION. For general $n$, even computing the minimum number of cuts required within an additive error of $n-1$ is still computationally hard as the following theorem shows.

**Theorem 3.** *For $n$ agents with linear utilities, it is NP-hard to compute the minimum number of cuts required in a consensus halving even if an additive error of $n-1$ is allowed.*

**Proof.** We reduce from the NP-hard problem PARTITION. Let $w_1, \dots, w_r$ be the integers that form a PARTITION instance. We construct a consensus halving instance $I$ with $n$ agents and a set of $n \cdot r$ items $M = \{(\ell, j) : \ell \in [n], j \in [r]\}$. Every agent values a distinct set of items according to the numbers $w_1, \dots, w_r$. Formally,

$$u_i((\ell, j)) = \begin{cases} w_j & \text{if } \ell = i; \\ 0 & \text{if } \ell \neq i \end{cases}$$

for all $i, \ell \in [n]$ and $j \in [r]$. It is easy to see that this instance has the following properties:

1. If $w_1, \dots, w_r$ can be partitioned into two sets of equal sum, then our instance $I$ admits a consensus halving using no cut.

2. If $w_1, \dots, w_r$ cannot be partitioned into two sets of equal sum, then any consensus halving of our instance $I$ uses at least $n$ cuts. This is because, in that case, for every agent $i \in N$, at least one of the items $(i,1), \dots, (i,r)$ must be cut.

As a result, in the first case, the minimum number of cuts is zero, whereas in the second case, this number is at least $n$. Thus, PARTITION reduces to the problem of computing the minimum number of cuts within an additive error of $n-1$. □

An immediate consequence of Theorem 3 is that the problem of computing a consensus halving that uses at most $n-1$ cuts more than the minimum number of cuts for the same instance is also NP-hard.

As our final remark of this section, consider utility functions that are again additively separable across items, but for which the utility of each item scales *quadratically* as opposed to linearly in the fraction of the item. That is, for a set $M'$ containing a fraction $x_j$ of item $j$, the utility of agent $i$ is given by $u_i(M') = \sum_{j \in M} x_j^2 \cdot u_i(j)$. Even though these utility functions appear different from the ones we consider so far, it turns out that the set of consensus halvings remains exactly the same. Indeed, a partition $(M_1, M_2)$ is a consensus halving under the quadratic functions if and only if

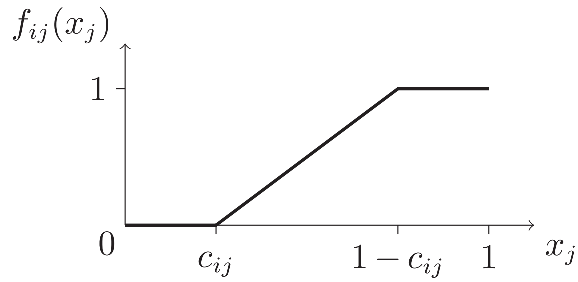
$$\sum_{j \in M} x_j^2 \cdot u_i(j) = \sum_{j \in M} (1 - x_j)^2 \cdot u_i(j) \qquad \forall i \in N.$$

Because $x_j^2 - (1-x_j)^2 = x_j - (1-x_j) = 2x_j - 1$, the preceding condition is equivalent to (1), so all of our results in this section apply to the quadratic functions as well.

## 3. Monotonic Utilities

Next, we turn our attention to utility functions that are no longer linear as in Section 2. We assume that the utilities are monotonic, meaning that the utility of an agent for a set of items cannot decrease upon adding any fraction of an item to the set. Our main result is that finding a consensus halving is computationally hard for such valuations; in fact, the hardness holds even when the utilities take on a specific structure that we call symmetric threshold. Symmetric-threshold utilities are additively separable across items and linear with symmetric thresholds within every item. Formally, the utility of agent $i$ for a fractional set of items $M'$ containing a fraction $x_j \in [0,1]$ of each item $j$ can be written as $u_i(M') = \sum_{j\in M} f_{ij}(x_j)\cdot u_i(j)$, where

$$f_{ij}(x_j) := \begin{cases} 0 & \text{if } x_j \le c_{ij}; \\ \dfrac{x_j - c_{ij}}{1-2c_{ij}} & \text{if } c_{ij} < x_j < 1 - c_{ij}; \\ 1 & \text{if } x_j \ge 1 - c_{ij}, \end{cases}$$




where $c_{ij} \in [0, 1/2)$ is the *threshold* or *cap* of agent $i$ for item $j$. Intuitively, symmetric-threshold utilities model settings in which having a small fraction of an item is the same as not having the item at all, whereas having a large fraction of the item is the same as having the whole item. The point at which this threshold behavior occurs is controlled by the cap $c_{ij}$, which can be different for every pair $(i,j) \in N \times M$. It is easy to see that the resulting utility functions are indeed monotonic. Note that, although general monotonic utility functions do not necessarily admit a concise representation (see the discussion preceding Theorem 6), symmetric-threshold utility functions can be described succinctly.

Even though symmetric-threshold utility functions are superficially similar to linear ones, we show that finding a consensus halving for such utilities is computationally hard. Recall that a partition $(M_1, M_2)$ is an $\varepsilon$-approximate consensus halving if $|u_i(M_1) - u_i(M_2)| \le \varepsilon \cdot u_i(M)$ for every agent $i$.

**Theorem 4.** *There exists a constant $\varepsilon > 0$ such that finding an $\varepsilon$-approximate consensus halving for $n$ agents with monotonic utilities that uses at most $n$ cuts is PPAD-hard even if all agents have symmetric-threshold utilities.*

**Proof.** We prove this result by reducing from a modified version of the generalized circuit problem. The generalized circuit problem is the main tool that has been used (implicitly or explicitly) to prove the hardness of computing Nash equilibria in various settings (Chen et al. [13], Daskalakis et al. [14], Rubinstein [42]). A generalized circuit is a generalization of an arithmetic circuit because it allows *cycles*, which means that, instead of a simple computation, the circuit now represents a constraint satisfaction problem. The version of the problem we use is different from the standard one in two aspects. First, instead of the domain $[0,1]$, we use $[-1,1]$, which is more adapted to the consensus halving problem. Second, we only allow the circuit to use three types of arithmetic gates. As we show, these modifications do not change the complexity of the problem.

Formally, we consider the following simplified generalized circuits.

**Definition 2.** A *simple* generalized circuit is a pair $(V, \mathcal{T})$, where $V$ is a set of nodes and $\mathcal{T}$ is a set of gates. Every gate $T \in \mathcal{T}$ is a five-tuple $T = (G, u_1, u_2, v, \zeta)$, where $G \in \{G_+, G_{\times-\zeta}, G_1\}$ is the type of gate; $u_1$, $u_2$ are the input nodes (if applicable); $\zeta \in (0,1]$ is the parameter (if applicable); and $v$ is the output node. In more detail,

- If $G = G_+$, then $u_1, u_2, v \in V$ (distinct) and $\zeta = nil$.
- If $G = G_{\times-\zeta}$, then $u_1, v \in V$ (distinct), $u_2 = nil$, and $\zeta \in (0,1]$.
- If $G = G_1$, then $u_1 = u_2 = \zeta = nil$ and $v \in V$.

We require that, for any two gates $T = (G, u_1, u_2, v, \zeta)$ and $T' = (G', u_1', u_2', v', \zeta')$ in $\mathcal{T}$ with $T \ne T'$, it holds that $v \ne v'$.

Before we proceed, let us introduce some notation. We let $T_{[-1,1]} : \mathbb{R} \to [-1,1]$ denote *truncation* to $[-1,1]$, that is, $T_{[-1,1]}(x) = \max\{-1, \min\{1, x\}\}$. Similarly, we also let $T_{[0,1]}$ denote truncation to $[0,1]$. Finally, we use the notation $x = y \pm z$ as a shorthand for $|x - y| \le z$.

**Definition 3.** Let $\varepsilon > 0$. The problem $\varepsilon$-SIMPLE-GCIRCUIT is defined as follows: given a simple generalized circuit $(V, \mathcal{T})$, find an assignment $\mathbf{x} : V \to [-1,1]$ that $\varepsilon$-approximately satisfies all the gates $T = (G, u_1, u_2, v, \zeta)$ in $\mathcal{T}$, namely,

- If $G = G_+$, then $\mathbf{x}[v] = T_{[-1,1]}(\mathbf{x}[u_1] + \mathbf{x}[u_2]) \pm \varepsilon$ (*addition*).
- If $G = G_{\times-\zeta}$, then $\mathbf{x}[v] = -\zeta \cdot \mathbf{x}[u_1] \pm \varepsilon$ (*multiplication by* $-\zeta$ *for* $\zeta \in (0,1]$).
- If $G = G_1$, then $\mathbf{x}[v] = 1 \pm \varepsilon$ (*constant* 1).

As mentioned earlier, it turns out that this modified version of the generalized circuit problem is also PPAD-hard. This can be proved by reducing from the standard $\varepsilon$-GCIRCUIT problem, which is shown to be PPAD-hard even for constant $\varepsilon$ by Rubinstein [42]. The idea is that these simple gates are enough to simulate all the gates in the standard version of the problem. Both problems are, in fact, PPAD-complete because they can be reduced to the problem of finding an approximate Brouwer fixed point, but here, we are only interested in the hardness.

**Lemma 1.** *There exists a constant $\varepsilon > 0$ such that the $\varepsilon$-SIMPLE-GCIRCUIT problem is PPAD-hard.*

The proof of Lemma 1 can be found in Appendix B.

Let $\widehat{\varepsilon} > 0$ be a constant for which the $\widehat{\varepsilon}$-SIMPLE-GCIRCUIT problem is PPAD-hard. We now show that the $\widehat{\varepsilon}$-SIMPLE-GCIRCUIT problem reduces to the problem of finding an $\varepsilon$-approximate consensus halving for $n$ agents with symmetric-threshold utilities that uses at most $n$ cuts.

Let $(V, \mathcal{T})$ be an instance of $\widehat{\varepsilon}$-SIMPLE-GCIRCUIT. Partition $V$ into four sets $V_0 \cup V_+ \cup V_\times \cup V_1$, where

- $V_0$ contains every node that is not the output of any gate in $\mathcal{T}$.
- $V_+$ contains every node that is the output of a $G_+$ gate in $\mathcal{T}$.
- $V_\times$ contains every node that is the output of a $G_{\times-\zeta}$ gate in $\mathcal{T}$.
- $V_1$ contains every node that is the output of a $G_1$ gate in $\mathcal{T}$.

We construct a consensus halving instance with $n = 2|V_+| + |V_\times| + |V_1|$ agents and $m = |V_0| + 2|V_+| + |V_\times| + |V_1| + 1$ items. For any node $v \in V_+ \cup V_\times \cup V_1$, let $i(v) \in N = [n]$ denote the corresponding agent, and for every $v \in V_+$, let $i'(v) \in N$ denote the second corresponding agent. For every $v \in V$, let $j(v) \in M = [m]$ denote the corresponding item, and for every $v \in V_+$, let $j'(v) \in M$ denote the second corresponding item. Finally, let $j^* \in M$ denote the single remaining item, which we call the *special item*.

It remains to specify the utility functions for the agents and the constant $\varepsilon > 0$. We see that, in any partition of $M$ into two fractional sets of items $(M_1, M_2)$, there is a simple way to associate a value $\text{val}(j) \in [-1,1]$ to every item $j \in M$. We pick the agents' utilities so that, in any $\varepsilon$-approximate consensus halving (with at most $n$ cuts), these values must satisfy the gate constraints in $\mathcal{T}$.

## 3.1. Value Encoding

Consider any partition of $M$ into two fractional sets of items $(M_1, M_2)$. Let $x_j \in [0,1]$ denote the fraction of item $j$ in $M_1$. This fraction $x_j \in [0,1]$ encodes a number $\text{val}(j) \in [-1,1]$ as follows:

$$\text{val}(j) = \begin{cases} -1 & \text{if } x_j \leq 1/3; \\ 6(x_j - 1/2) & \text{if } 1/3 < x_j < 2/3; \\ 1 & \text{if } x_j \geq 2/3. \end{cases}$$

In other words, $\text{val}(j) = T_{[-1,1]}(6x_j - 3)$.

The main idea of the reduction is that the value $\mathbf{x}[v]$ of node $v \in V$ is given by $\text{val}(j(v))$. Next, we show how to pick the utility functions in order to enforce the gate constraints in $\mathcal{T}$. In the following construction, we assume that $\varepsilon \leq 1/10$; the exact value of $\varepsilon$ is picked at the end.

## 3.2. $G_{\times-\zeta}$ Gates

For any gate $(G_{\times-\zeta}, u_1, nil, v, \zeta) \in \mathcal{T}$, where $u_1 \in V \backslash \{v\}$, $v \in V_\times$ and $\zeta \in (0,1]$, we do the following. Let $j_1 = j(u_1)$, $j = j(v)$ and $i = i(v)$. We want to ensure that, in any solution to $\varepsilon$-approximate consensus halving, we have $\text{val}(j) = -\zeta \cdot \text{val}(j_1) \pm \widehat{\varepsilon}$. To achieve this, we define the symmetric-threshold utility function of agent $I$ as follows. For any item $\ell \notin \{j_1, j\}$, we let $u_i(\ell) = 0$ and $c_{i\ell} = 0$. We let $u_i(j) = 1/\zeta$ and $c_{ij} = 0$. For $j_1$, we use what we call a *standard input utility function*, which is defined as follows: $u_i(j_1) = 1/3$ and $c_{ij_1} = 1/3$. Note that $u_i(M) = 1/3 + 1/\zeta$.

Consider any $\varepsilon$-approximate consensus halving $(M_1, M_2)$. Then, it must hold that $u_i(M_1) = u_i(M_2) \pm \varepsilon \cdot u_i(M)$. First of all, because $u_i(j) > u_i(M \backslash \{j\}) + \varepsilon \cdot u_i(M)$ and by monotonicity, this implies that item $j$ must be fractional in the partition $(M_1, M_2)$, that is, $x_j \in (0,1)$. Furthermore, we must have

$$f_{ij_1}(x_{j_1}) u_i(j_1) + f_{ij}(x_j) u_i(j) = f_{ij_1}(1 - x_{j_1}) u_i(j_1) + f_{ij}(1 - x_j) u_i(j) \pm \varepsilon \cdot u_i(M).$$

Because $f_{i\ell}(1 - x) = 1 - f_{i\ell}(x)$ for any $x \in [0,1]$ and $\ell \in M$, this equation can be rewritten as

$$(2f_{ij}(x_j) - 1) u_i(j) = -(2f_{ij_1}(x_{j_1}) - 1) u_i(j_1) \pm \varepsilon \cdot u_i(M).$$

By noting that $f_{i\ell}(x) = T_{[0,1]}((x - c_{i\ell})/(1 - 2c_{i\ell}))$, we obtain

$$(2x_j - 1) \cdot (1/\zeta) = -(2T_{[0,1]}(3x_{j_1} - 1) - 1) \cdot (1/3) \pm \varepsilon \cdot u_i(M).$$

Finally, by observing that $2T_{[0,1]}(3x_{j_1} - 1) - 1 = T_{[-1,1]}(2(3x_{j_1} - 1) - 1) = T_{[-1,1]}(6x_{j_1} - 3) = \mathrm{val}(j_1)$, we obtain

$$(6x_j - 3) = -\zeta \cdot \mathrm{val}(j_1) \pm 3\zeta\varepsilon \cdot u_i(M).$$

Now, this yields

$$\mathrm{val}(j) = T_{[-1,1]}(6x_j - 3) = T_{[-1,1]}(-\zeta \cdot \mathrm{val}(j_1)) \pm 3\zeta\varepsilon \cdot u_i(M) = -\zeta \cdot \mathrm{val}(j_1) \pm 4\varepsilon,$$

where we use the fact that $-\zeta \cdot \mathrm{val}(j_1) \in [-1,1]$, $u_i(M) = 1/3 + 1/\zeta$ and $\zeta \le 1$. Thus, as long as $4\varepsilon \le \widehat{\varepsilon}$, this construction correctly enforces the gate constraint.

### 3.3. $G_1$ Gates

For any gate $(G_1, nil, nil, v, nil) \in \mathcal{T}$, where $v \in V_1$, we do the following. Let $j = j(v)$ and $i = i(v)$. We use the same construction as for $G_{\times-\zeta}$ gates with $j_1 = j^*$ (the special item) and $\zeta = 1$. By the same arguments, it follows that, in any $\varepsilon$-approximate solution, it must hold that $\mathrm{val}(j) = -\mathrm{val}(j^*) \pm 4\varepsilon$, and item $j$ must be fractional, that is, $x_j \in (0,1)$. Thus, as long as $4\varepsilon \le \widehat{\varepsilon}$ and $\mathrm{val}(j^*) = -1$, this correctly enforces the gate constraint.

### 3.4. $G_+$ Gates

For any gate $(G_+, u_1, u_2, v, nil) \in \mathcal{T}$, where $u_1 \in V\backslash\{v\}$, $u_2 \in V\backslash\{v, u_1\}$ and $v \in V_+$, we do the following. Let $j_1 = j(u_1)$, $j_2 = j(u_2)$, $j = j(v)$, and $j' = j'(v)$. We are going to ensure that $\mathrm{val}(j') = -T_{[-1,1]}(\mathrm{val}(j_1) + \mathrm{val}(j_2)) \pm \widehat{\varepsilon}/2$ and $\mathrm{val}(j) = -\mathrm{val}(j') \pm \widehat{\varepsilon}/2$. Together, these two constraints enforce the gate constraint. The second constraint can easily be enforced by using the same construction as for $G_{\times-\zeta}$ with $j_1 = j'(v)$, $j = j(v)$, $i = i(v)$ and $\zeta = 1$. By the same arguments, this yields an error of at most $\widehat{\varepsilon}/2$ as long as $8\varepsilon \le \widehat{\varepsilon}$ and ensures that item $j$ is fractional.

To enforce the first constraint, we define the utilities of agent $i' = i'(v)$ as follows. For any item $\ell \notin \{j_1, j_2, j'\}$, we let $u_{i'}(\ell) = 0$ and $c_{i'\ell} = 0$. We let $u_{i'}(j') = 1$ and $c_{i'j'} = 0$. For $j_1$ and $j_2$, we use the standard input utility function as defined earlier. Note that $u_{i'}(M) = 5/3$.

Consider any $\varepsilon$-approximate consensus halving ($M_1$, $M_2$). Then, it must hold that $u_{i'}(M_1) = u_{i'}(M_2) \pm \varepsilon \cdot u_{i'}(M)$. First of all, because $u_{i'}(j') > u_{i'}(M\backslash\{j'\}) + \varepsilon \cdot u_{i'}(M)$ and by monotonicity, this implies that item $j'$ must be fractional in the partition ($M_1$, $M_2$), that is, $x_{j'} \in (0,1)$. Furthermore, by the same arguments as for $G_{\times-\zeta}$ gates, we obtain that

$$6x_{j'} - 3 = -\mathrm{val}(j_1) - \mathrm{val}(j_2) \pm 3\varepsilon \cdot u_i(M).$$

Because $\mathrm{val}(j') = T_{[-1,1]}(6x_{j'} - 3)$, it follows that $\mathrm{val}(j') = -T_{[-1,1]}(\mathrm{val}(j_1) + \mathrm{val}(j_2)) \pm 5\varepsilon$. Thus, this constraint is correctly enforced as long as $10\varepsilon \le \widehat{\varepsilon}$.

We are now ready to complete the proof. Set $\varepsilon = \widehat{\varepsilon}/10$. Consider any $\varepsilon$-approximate consensus halving ($M_1$, $M_2$) that uses at most $n$ cuts. We claim that letting $\mathbf{x}[v] = \mathrm{val}(j(v))$ for all $v \in V$ yields a solution to the $\widehat{\varepsilon}$-SIMPLE-GCIRCUIT instance. Indeed, by construction, all gates of type $G_+$ and $G_{\times-\zeta}$ are correctly enforced. For gates of type $G_1$, they are correctly enforced if $\mathrm{val}(j^*) = -1$, which we now prove. Note that, in our construction, we have ensured that, for every $v \in V_+ \cup V_\times \cup V_1$, item $j(v)$ must be fractional, and for every $v \in V_+$, item $j'(v)$ must also be fractional. Because these $2|V_+| + |V_\times| + |V_1| = n$ items are fractional, and we use at most $n$ cuts, this means that all other items are *not* fractional. In particular, $j^*$ is not fractional, that is, $x_{j^*} \in \{0,1\}$. Without loss of generality, assume that $x_{j^*} = 0$ (if $x_{j^*} = 1$, then swap the roles of $M_1$ and $M_2$). It follows that $\mathrm{val}(j^*) = -1$. □

## 4. Connections to Agreeable Sets

We now present some implications of results from consensus halving on the setting of computing agreeable sets. Let us first formally define the agreeable set problem, introduced by Manurangsi and Suksompong [33].[8] As in consensus halving, there is a set $N$ of $n$ agents and a set $M$ of $m$ items. Agent $i$ has a monotonic utility function $u_i$ over *nonfractional* sets of items, in which we assume the normalization $u_i(\emptyset) = 0$; this corresponds to a set function. Note that, because we are only concerned with discrete items, we use the term "additive" instead of additively separable in this section.

**Definition 4.** A subset of items $M' \subseteq M$ is said to be agreeable to agent $i$ if $u_i(M') \ge u_i(M\backslash M')$.

As one of their main results, Manurangsi and Suksompong [33] show that, for any $n$ and $m$, there exists a set of at most $\min\{\lfloor \frac{m+n}{2} \rfloor, m\}$ items that is agreeable to all agents, and this bound is tight. Their proof relies on a graph-theoretic statement often referred to as "Kneser's conjecture," which specifies the chromatic number for a particular class of graphs called Kneser graphs. Here, we present a short alternative proof that works by arranging the items on a line in arbitrary order, applying consensus halving, and rounding the resulting fractional partition. As a bonus, our proof yields an agreeable set that is composed of at most $\lfloor n/2 \rfloor + 1$ blocks on the line.

**Theorem 5** (Manurangsi and Suksompong [33]). *For $n$ agents with monotonic utilities, there exists a subset $M' \subseteq M$ such that*

$$|M'| \le \min\left\{\left\lfloor \frac{m+n}{2} \right\rfloor, m\right\}$$

*and $M'$ is agreeable to all agents.*

**Proof.** Let $s = \lfloor \frac{m+n}{2} \rfloor$. If $s \ge m$, the entire set of items $M$ has size $m = \min\{s, m\}$ and is agreeable to all agents because of monotonicity, so we may assume that $s \le m$. Arrange the items on a line in arbitrary order and extend the utility functions of the agents to fractional sets of items in a continuous and monotonic fashion.[9] Consider a consensus halving with respect to the extended utilities that uses at most $n$ cuts on the line; some of the cuts may cut through items, whereas the remaining cuts are between adjacent items. Let $r \le n$ be the number of items that are cut by at least one cut. Without loss of generality, assume that the first part $M'$ contains no more full items than the second part $M''$, so $M'$ contains at most $\lfloor \frac{m-r}{2} \rfloor$ full items. By moving all cut items from $M''$ to $M'$ in their entirety, $M'$ contains at most $\lfloor \frac{m-r}{2} \rfloor + r = \lfloor \frac{m+r}{2} \rfloor \le s$ items. Because we start with a consensus halving and only move fractional items from $M''$ to $M'$, we have that $M'$ is agreeable to all agents. Moreover, one can check that $M'$ is composed of at most $\lceil \frac{n+1}{2} \rceil = \lfloor \frac{n}{2} \rfloor + 1$ blocks on the line. □

In light of Theorem 5, an important question is how efficiently we can compute an agreeable set whose size matches the worst-case bound. Manurangsi and Suksompong [33] address this question by providing a polynomial-time algorithm for two agents with monotonic utilities and three agents with "responsive" utilities, a class that lies between additive and monotonic utilities. They leave the complexity for higher numbers of agents as an open question and conjecture that the problem is hard even when the number of agents is a larger constant. We show that this is, in fact, not the case: the problem can be solved efficiently for any number of agents with additive utilities as well as for any *constant* number of agents with monotonic utilities. Note that, because the input of the problem for monotonic utilities can involve an exponential number of values (even for constant $n$) and, consequently, may not admit a succinct representation, we assume a "utility oracle model" in which the algorithm is allowed to query the utility $u_i(M')$ for any $i \in N$ and $M' \subseteq M$.

**Theorem 6.** *There exists a polynomial-time algorithm that computes a set containing at most $\min\{\lfloor \frac{m+n}{2} \rfloor, m\}$ items that is agreeable to all agents for each of the following two cases:*

i. *All agents have additive utilities.*

ii. *All agents have monotonic utilities, and the number of agents is constant (assuming access to a utility oracle).*

**Proof.** Similarly to Theorem 5, if $n \ge m$, we can simply include all items in our set, so we may focus on the case $n \le m$. For (i), we first use our polynomial-time algorithm from Theorem 1 to find a consensus halving and then compute an agreeable set of size at most $\lfloor \frac{m+n}{2} \rfloor$ by rounding the consensus halving as in the proof of Theorem 5.

Next, consider (ii). Recall that, for any ordering of the items on a line, Theorem 5 guarantees the existence of an agreeable set of size at most $\lfloor \frac{m+n}{2} \rfloor$ involving no more than $n$ cuts on the line. Fix an ordering of the items; we perform a brute-force search over all (nonfractional) partitions involving at most $n$ cuts with respect to the ordering. For $t \in [n]$, there are $O(m^t)$ ways to place $t$ cuts, and for each way, we have two candidate sets to check: one including the leftmost item and one not including it. A candidate set is valid if and only if it has size at most $\lfloor \frac{m+n}{2} \rfloor$ and is agreeable to all agents. Hence, the brute-force search runs in time $\sum_{t=1}^{n} O(m^t) = O(n \cdot m^n) = O(m^n)$, which is polynomial because $n$ is constant. □

## 5. Consensus *k*-Splitting

In this section, we address two important generalizations of consensus halving, both of which are mentioned by Simmons and Su [50]. In consensus splitting, instead of dividing the items into two equal parts, we want to divide them into two parts so that all agents agree that the split satisfies some given ratio, say two to one. In consensus $1/k$-division, we want to divide the items into $k$ parts that all agents agree are equal. We consider a problem that generalizes both of these problems at once.

**Definition 5.** Let $\alpha_1, \ldots, \alpha_k > 0$ be real numbers such that $\alpha_1 + \cdots + \alpha_k = 1$. A consensus $k$-splitting with ratios $\alpha_1, \ldots, \alpha_k$ is a partition of $M$ into $k$ fractional sets of items $M_1, \ldots, M_k$ such that

$$\frac{u_i(M_1)}{\alpha_1} = \frac{u_i(M_2)}{\alpha_2} = \cdots = \frac{u_i(M_k)}{\alpha_k} \qquad \forall i \in N.$$

When the ratios are clear from context, we simply refer to such a partition as a consensus $k$-splitting.

As in Section 2, we assume that the utility functions are linear, in which case our desired condition is equivalent to $u_i(M_\ell) = \alpha_\ell \cdot u_i(M)$ for all $i \in N$ and $\ell \in [k]$.

Whereas there is no reason to cut an item more than once in consensus halving, one may sometimes wish to cut the same item multiple times in consensus $k$-splitting in order to split the item across three or more parts. Hence, even though the number of cuts made is always at least the number of items cut, the two quantities are not necessarily the same in consensus $k$-splitting. If there are $n$ items and each agent only values a single distinct item, then it is clear that we already need to make $(k-1)n$ cuts for any ratios $\alpha_1, \dots, \alpha_k$, in particular, $k-1$ cuts for each item. Nevertheless, it could still be that, for some ratios, it is always possible to achieve a consensus $k$-splitting by cutting fewer than $(k-1)n$ items. We show that this is not the case: for any set of ratios, cutting $(k-1)n$ items is necessary in the worst case.

**Theorem 7.** *For any ratios $\alpha_1, \dots, \alpha_k > 0$, there exists an instance with linear utilities in which any consensus k-splitting with these ratios cuts at least $(k-1)n$ items.*

**Proof.** Fix $\alpha_1, \dots, \alpha_k > 0$. We construct an instance such that each agent $i$ has utility $1/b$ for each of the $b$ items in a set $B_i$, where $b$ is an integer that we choose later, and utility zero for every other item. The sets $B_1, \dots, B_n$ are pairwise disjoint. Note that $u_i(M) = u_i(B_i) = 1$ for every $i$. It suffices to choose $b$ such that at least $k-1$ items in each set $B_i$ must be cut in any consensus $k$-splitting with ratios $\alpha_1, \dots, \alpha_k$. By symmetry, we may focus on the first agent and the corresponding set $B_1$.

For any real number $x$, denote by $\lfloor x \rfloor$ its floor function, and let $\{x\} = x - \lfloor x \rfloor$. We choose $b$ such that

$$\{\alpha_1 b\} + \{\alpha_2 b\} + \cdots + \{\alpha_k b\} > k-2. \tag{2}$$

To see why this is sufficient, observe that each *uncut* item must belong to one of the $k$ parts in its entirety. The number of uncut items in $B_1$ is, therefore, at most

$$\left\lfloor \frac{\alpha_1}{1/b} \right\rfloor + \cdots + \left\lfloor \frac{\alpha_k}{1/b} \right\rfloor = \lfloor \alpha_1 b \rfloor + \cdots + \lfloor \alpha_k b \rfloor,$$

meaning that the number of cut items in $B_1$ is at least

$$\begin{aligned} b - (\lfloor \alpha_1 b \rfloor + \cdots + \lfloor \alpha_k b \rfloor) &= (\alpha_1 b + \cdots + \alpha_k b) - (\lfloor \alpha_1 b \rfloor + \cdots + \lfloor \alpha_k b \rfloor) \\ &= (\alpha_1 b - \lfloor \alpha_1 b \rfloor) + \cdots + (\alpha_k b - \lfloor \alpha_k b \rfloor) \\ &= \{\alpha_1 b\} + \cdots + \{\alpha_k b\} \\ &> k-2, \end{aligned}$$

where the first equality follows from $\alpha_1 + \cdots + \alpha_k = 1$. Because $b, \lfloor \alpha_1 b \rfloor, \dots, \lfloor \alpha_k b \rfloor$ are all integers, this implies that at least $k-1$ items in $B_1$ must be cut.

It remains to show the existence of $b$ for which (2) is satisfied. Let $s$ be an integer such that

$$s > \max\left\{k, \frac{1}{\alpha_1}, \cdots, \frac{1}{\alpha_k}, \frac{1}{1-\alpha_1}, \cdots, \frac{1}{1-\alpha_k}\right\}.$$

Divide the interval $[0,1]$ into subintervals of length at most $1/s$ each. By the pigeonhole principle, there exist positive integers $p$, $q$ such that $q \geq p+2$, and $\{\alpha_i p\}$ and $\{\alpha_i q\}$ fall in the same subinterval for every $i \in [k]$. Letting $c = q - p$, we have that for each $i \in [k]$, either $\{\alpha_i c\} < 1/s$ or $\{\alpha_i c\} > 1 - 1/s$.

Take $b = c - 1 \geq 1$. From our choice of $s$, we have $1/s < \alpha_i < 1 - 1/s$ for all $i \in [k]$. Thus, for each $i$, if $\{\alpha_i c\} < 1/s$, then $\{\alpha_i c\} < \alpha_i$, whereas if $\{\alpha_i c\} > 1 - 1/s$, then $\{\alpha_i c\} > \alpha_i$. In either case, we have $\{\alpha_i b\} = \{\alpha_i c - \alpha_i\} > 1 - 1/s - \alpha_i$, so

$$\{\alpha_1 b\} + \cdots + \{\alpha_k b\} > k - k/s - (\alpha_1 + \cdots + \alpha_k) > k-2,$$

where we use the assumption that $s > k$. Hence, (2) is satisfied, and the proof is complete. □

Next, we show that computing a consensus $k$-splitting with at most $(k-1)n$ cuts can be done efficiently using a generalization of our algorithm for consensus halving (Theorem 1). Note that such a splitting also cuts at most $(k-1)n$ items.

**Theorem 8.** *For n agents with linear utilities and ratios $\alpha_1, \dots, \alpha_k$, there is a polynomial-time algorithm that computes a consensus k-splitting with these ratios using at most $(k-1) \cdot \min\{n, m\}$ cuts.*

**Proof.** Let us start with the case $k = 2$, which can then be used as a subroutine for the case $k > 2$. Our algorithm for consensus two-splitting generalizes the consensus halving algorithm in Theorem 1, so we only highlight the differences. To find a consensus two-splitting with ratios $\alpha_1, \alpha_2$, the only change to the algorithm in Theorem 1 is that we initialize $x_1 = \cdots = x_m = \alpha_1$ and let $S$ be the set of $n$ equations $\sum_{j\in M}(y_j - \alpha_1)\cdot u_i(j)$ for $i \in N$. By analogous arguments as in Theorem 1, this modified algorithm produces a consensus two-splitting with ratios $\alpha_1, \alpha_2$ in polynomial time and uses at most $\min\{n, m\}$ cuts.

We now move on to the case $k > 2$. In this case, we simply apply the consensus two-splitting algorithm successively, each time producing one additional part at the expense of at most $\min\{n, m\}$ cuts. This is stated more precisely:

1. Let $M_{\text{remaining}} = M$.
2. For $\ell = 1, \ldots, k-1$,
   a. $(M_\ell, M_{\text{remaining}}) \leftarrow$ consensus two-splitting of $M_{\text{remaining}}$ with ratios $\frac{\alpha_\ell}{\alpha_\ell + \cdots + \alpha_n}, \frac{\alpha_{\ell+1} + \cdots + \alpha_n}{\alpha_\ell + \cdots + \alpha_n}$.
3. Output $(M_1, \ldots, M_{k-1}, M_{\text{remaining}})$.

It is clear that the output is a consensus $k$-splitting with ratios $\alpha_1, \ldots, \alpha_k$ and that the algorithm runs in polynomial time. Finally, observe that, each time we apply the consensus two-splitting algorithm, if there are $m'$ items left, we additionally use at most $\min\{n, m'\} \le \min\{n, m\}$ cuts. As a result, the total number of cuts is at most $(k-1)\cdot \min\{n, m\}$ as desired. □

As in Theorem 1, our algorithm does not require the nonnegativity assumption on the utilities and, therefore, works for combinations of goods and chores.

When the items lie on a line, there is always a consensus halving that makes at most $n$ cuts on the line and, therefore, cuts at most $n$ items; this matches the upper bound on the number of items cut in the absence of a linear order. Theorem 8 shows that the bound $n$ continues to hold for consensus splitting into two parts with any ratios. As we show next, however, this bound is no longer achievable for some ratios with ordered items, thereby demonstrating another difference that the lack of linear order makes.[10]

**Theorem 9.** *Let $n \ge 2$, $k = 2$ and $(\alpha_1, \alpha_2) = (\frac{1}{n}, \frac{n-1}{n})$. There exists an instance such that the n agents have linear utilities, the items lie on a line, and any consensus k-splitting with ratios $\alpha_1$ and $\alpha_2$ makes at least $2n-4$ cuts on the line.*

**Proof.** We discretize a slight modification of an instance used by Stromquist and Woodall [51] to show a lower bound on the number of cuts when the resource is represented by a one-dimensional circle. Suppose that there are $n^2 - 1$ "primary items," which we label as $1, 2, \ldots, n^2 - 1$ according to their linear order. Moreover, there are $n^2 - 2$ "secondary items," one between every adjacent pair of primary items. The utilities of the agents are as follows:

- For $i \in [n-1]$, agent $i$ has utility $\frac{1}{n+1}$ for each of the $n+1$ primary items $i, i+(n-1), \ldots, i+n(n-1)$ and utility zero for all secondary items.
- Agent $n$ has value $\frac{1}{n^2-2}$ for each secondary item and value zero for all primary items.

Note that $u_i(M) = 1$ for all $i$. Let $M'$ be a fractional set of items for which all agents have utility $1/n$. Because each agent $i \in [n-1]$ has utility $\frac{1}{n+1}$ for a primary item, $M'$ must contain a positive fraction of at least two primary items that the agent values. These items are disjoint for different agents, so $M'$ necessarily contains a positive fraction of at least $2n-2$ primary items. On the other hand, the utility function of agent $n$ implies that $M'$ can contain at most $\lfloor \frac{1/n}{1/(n^2-2)} \rfloor = n-1$ entire secondary items.

Suppose that $M'$ is composed of $r$ nonadjacent intervals $I_1, \ldots, I_r$. Notice that, for any interval $I$ on the line, if the interval contains a positive fraction of $t_1(I)$ primary items along with $t_2(I)$ *entire* secondary items, then $t_1(I) \le t_2(I) + 1$. Hence, we have

$$2n - 2 \le \sum_{i=1}^{r} t_1(I_i) \le \sum_{i=1}^{r} t_2(I_i) + r \le n - 1 + r,$$

implying that $r \ge n-1$. This means that the consensus two-splitting with $M'$ as one part involves at least $2(n-1) = 2n-2$ cuts, possibly including endpoints of the line. At most two of these cuts can correspond to endpoints, so the number of cuts made is at least $2n-4$ as desired. □

## 6. Probabilistic Results

In this section, instead of considering consensus halving and consensus $k$-splitting from a worst-case perspective as we have done so far, we examine these problems from an average-case perspective. We assume throughout the section that the agents have linear utilities.

First, Theorem 3 implies that there is no hope of finding a consensus halving with the minimum number of cuts or even a nontrivial approximation thereof in polynomial time provided that P ≠ NP. Nevertheless, we show that instances that admit a consensus halving with fewer than $n$ cuts are rare: if the utilities are drawn independently at random from probability distributions, then it is almost surely the case that any consensus halving needs at least $n$ cuts. We say that a distribution is *nonatomic* if it does not put a positive probability on any single point.

**Theorem 10.** *Suppose that, for each $i \in N$ and $j \in M$, the utility $u_i(j)$ is drawn independently from a nonatomic distribution $\mathcal{D}_{i,j}$. Then, with probability one, every consensus halving uses at least* $\min\{n, m\}$ *cuts.*

**Proof.** The high-level idea is to show that, if there are fewer than $\min\{n, m\}$ cuts, then a certain utility $u_i(j)$ needs to take on a specific value; this event occurs with probability zero because the distribution $\mathcal{D}_{i,j}$ is nonatomic.

Let $m_{\text{cut}} = \min\{n, m\} - 1$. Recall that a consensus halving corresponds to a tuple $(x_1, \ldots, x_m) \in [0,1]^m$ for which Constraint (1) is satisfied and that item $j$ is cut if and only if $x_j \notin \{0,1\}$. As a result, from the union bound, it suffices to show that, for any fixed $M_{\text{cut}} \subseteq M$ of size $m_{\text{cut}}$, we have

$$\Pr[\exists (x_1, \ldots, x_m) \in [0,1]^m \text{ that satisfies (1) and } x_j \in \{0,1\} \text{ for all } j \notin M_{\text{cut}}] = 0. \tag{3}$$

For notational convenience, we only show that (3) holds for $M_{\text{cut}} = \{1, \ldots, m_{\text{cut}}\}$; because of symmetry, the same bound also holds for every $M_{\text{cut}} \subseteq M$ of size $m_{\text{cut}}$.

To show (3) for $M_{\text{cut}} = \{1, \ldots, m_{\text{cut}}\}$, we may apply the union bound again to derive

$$\begin{aligned}&\Pr[\exists (x_1, \ldots, x_m) \in [0,1]^m \text{ that satisfies (1) and } x_j \in \{0,1\} \text{ for all } j \in \{m_{\text{cut}}+1, \ldots, m\}]\\ &\le \sum_{t_{m_{\text{cut}}+1}, \ldots, t_m \in \{0,1\}} \Pr[\exists x_1, \ldots, x_{m_{\text{cut}}} \in [0,1] \text{ such that } (x_1, \ldots, x_{m_{\text{cut}}}, t_{m_{\text{cut}}+1}, \ldots, t_m) \text{ satisfies (1)}].\end{aligned}$$

Hence, it suffices to show that, for any fixed $t_{m_{\text{cut}}+1}, \ldots, t_m \in \{0,1\}$, we have

$$\Pr[\exists x_1, \ldots, x_{m_{\text{cut}}} \in [0,1] \text{ such that } (x_1, \ldots, x_{m_{\text{cut}}}, t_{m_{\text{cut}}+1}, \ldots, t_m) \text{ satisfies (1)}] = 0.$$

To see that this is the case, consider any fixed values of $u_i(j)$ for all $i \in N, j \in M_{\text{cut}}$; we show that the preceding probability is zero over the randomness of the utilities $u_i(j)$ for $i \in N, j \notin M_{\text{cut}}$. We may rearrange Constraint (1) as

$$\sum_{j \in M_{\text{cut}}} u_i(j) \cdot x_j = \frac{1}{2} \sum_{j \in M_{\text{cut}}} u_i(j) + \sum_{j \notin M_{\text{cut}}} \left(\frac{1}{2} - t_j\right) \cdot u_i(j) \qquad \forall i \in N. \tag{4}$$

Now, because there are $n$ linear equations and only $m_{\text{cut}} < n$ variables $x_1, \ldots, x_{m_{\text{cut}}}$, the coefficient vectors $(u_1(1), \ldots, u_1(m_{\text{cut}})), \ldots, (u_n(1), \ldots, u_n(m_{\text{cut}}))$ must be linearly dependent. In other words, there exists $(a_1, \ldots, a_n) \neq (0, \ldots, 0)$ such that

$$\sum_{i \in N} a_i \cdot u_i(j) = 0 \qquad \forall j \in M_{\text{cut}}.$$

Hence, by taking the corresponding linear combination of (4), we have

$$\begin{aligned}0 &= \sum_{j \in M_{\text{cut}}} x_j \left(\sum_{i \in N} a_i \cdot u_i(j)\right)\\ &= \sum_{i \in N} a_i \left(\sum_{j \in M_{\text{cut}}} x_j \cdot u_i(j)\right)\\ &= \sum_{i \in N} a_i \left(\frac{1}{2} \cdot \sum_{j \in M_{\text{cut}}} u_i(j) + \sum_{j \notin M_{\text{cut}}} \left(\frac{1}{2} - t_j\right) \cdot u_i(j)\right).\end{aligned}$$

From $(a_1, \ldots, a_n) \neq (0, \ldots, 0)$, there exists $i^* \in N$ such that $a_{i^*} \neq 0$. Moreover, because $m_{\text{cut}} < m$, we have $m \notin M_{\text{cut}}$. This equality, therefore, implies that

$$\begin{aligned}u_{i^*}(m) = \frac{1}{\left(t_m - \frac{1}{2}\right)} &\left(\sum_{\substack{i \in N\\ i \neq i^*}} \frac{a_i}{a_{i^*}} \left(\frac{1}{2} \cdot \sum_{j \in M_{\text{cut}}} u_i(j) + \sum_{j \notin M_{\text{cut}}} \left(\frac{1}{2} - t_j\right) \cdot u_i(j)\right)\right.\\ &\left. + \left(\frac{1}{2} \cdot \sum_{j \in M_{\text{cut}}} u_{i^*}(j) + \sum_{\substack{j \notin M_{\text{cut}}\\ j \neq m}} \left(\frac{1}{2} - t_j\right) \cdot u_{i^*}(j)\right)\right),\end{aligned}$$

where $t_m - 1/2$ is nonzero because $t_m \in \{0,1\}$. Because $\mathcal{D}_{i^*,m}$ is nonatomic and the utilities are drawn independently, the equality occurs with probability zero, which implies that

$$\Pr[\exists x_1, \cdots, x_{m_{\text{cut}}} \in [0,1] \text{ such that } (x_1, \ldots, x_{m_{\text{cut}}}, t_{m_{\text{cut}}+1}, \ldots, t_m) \text{ satisfies (1)}] = 0.$$

As discussed, this, in turn, implies that the probability that there is a consensus halving with at most $m_{\text{cut}}$ cuts is zero, concluding our proof. □

We now comment on the necessity of the two distributional assumptions in Theorem 10.

• Nonatomicity condition: Suppose $n = 1$ and $\mathcal{D}_{1,j}$ is the Bernoulli distribution with $p = 1/2$ for all $j \in M$; that is, $u_1(j) = 0$ and $u_1(j) = 1$ with probability $1/2$ each. Then, the minimum number of cuts is one if $u_i(j) = 1$ for an odd number of $j$ and zero otherwise; the probability that each event occurs is $1/2$.

• Independence condition: Suppose all agents have the same utility function; that is, the dependence between the utilities is such that $u_1(j) = \ldots = u_n(j)$ for all $j \in [m]$. In this case, it is clear that no more than one cut is needed regardless of $n$ and $m$.

Theorem 10 shows that, in a random consensus halving instance, any solution almost surely uses at least the worst-case number of cuts $\min\{n, m\}$. One might consequently expect that an analogous statement holds for consensus $k$-splitting with $(k-1)\cdot\min\{n, m\}$ cuts almost always being required. However, we show that this is not true: even in the simple case in which $n = 1$ and the agent's utilities are drawn from the uniform distribution over $[0,1]$, it is likely that we only need to make one cut (instead of $k-1$) for large $m$.

**Theorem 11.** *Let $n = 1$ and suppose that the agent's utility for each item is drawn independently from the uniform distribution on $[0,1]$. For any ratios $\alpha_1, \ldots, \alpha_k > 0$ with probability approaching one as $m \to \infty$, there exists a consensus $k$-splitting with these ratios using at most one cut. Moreover, there is a polynomial-time algorithm that computes such a solution.*

In what follows, we denote the agent's utility function by $u$ and say that an event happens "with high probability" if the probability that it happens approaches one as $m \to \infty$. The proof of Theorem 11 proceeds by identifying a simple (deterministic) condition that guarantees a solution cutting only a single item; this is done in Lemma 2. Then, we show that this condition is satisfied with high probability.

**Lemma 2.** *Suppose that there is a single agent. Let $j^* := \arg\max_j u(j)$ denote a most preferred item and let $M_{\text{low-utility}} := \left\{ j \in M \,|\, u(j) \le \frac{1}{k}\cdot u(j^*)\right\}$ denote the set of items whose utility is less than $1/k$ times the utility of $j^*$. For any ratios $\alpha_1, \ldots, \alpha_k > 0$, if $\sum_{j\in M_{\text{low-utility}}} u(j) \ge k\cdot u(j^*)$, then there is a consensus $k$-splitting with these ratios that cuts only $j^*$. Moreover, there is a polynomial-time algorithm that computes such a solution.*

**Proof.** For each $\ell \in [k]$, let $w_\ell := \alpha_\ell \cdot \left(\sum_{j\in M} u(j)\right)$ be the "target utility" for part $\ell$ of the partition. Consider the following greedy algorithm.

- Let $P_1 = \cdots = P_k = \emptyset$.
- Let $M^0 = M \setminus \{j^*\}$ and $j^{\max} = \arg\max_{j\in M^0} u(j)$.
- While there exists $\ell \in [k]$ such that $u(P_\ell \cup \{j^{\max}\}) \le w_\ell$:
  - — Add $j^{\max}$ to $P_\ell$.
  - — Remove $j^{\max}$ from $M^0$. If $M^0 = \emptyset$, terminate. Else, update $j^{\max} = \arg\max_{j\in M^0} u(j)$.

The algorithm clearly runs in polynomial time. We claim that it terminates with $M^0 = \emptyset$ provided that $\sum_{j\in M_{\text{low-utility}}} u(j) \ge k\cdot u(j^*)$. This implies the statement of the lemma because it would then suffice to split only item $j^*$.

Suppose for the sake of contradiction that $M^0 \ne \emptyset$ at the end of the execution. Consider the following two cases based on whether $j^{\max}$ at termination belongs to $M_{\text{low-utility}}$.

Case 1: $j^{\max} \notin M_{\text{low-utility}}$. Because the algorithm terminates, it must be that $u(P_\ell) > w_\ell - u(j^{\max}) \ge w_\ell - u(j^*)$ for each $\ell$. Summing this over $\ell \in [k]$, we get

$$u(P_1 \cup \ldots \cup P_k) > w_1 + \cdots + w_k - k\cdot u(j^*) = u(M) - k\cdot u(j^*).$$

On the other hand, because $j^{\max} \notin M_{\text{low-utility}}$, it must be that $M_{\text{low-utility}}$ is disjoint from $P_1 \cup\cdots\cup P_k$. As a result, we have

$$u(P_1 \cup \ldots \cup P_k) \le u(M) - \sum_{j\in M_{\text{low-utility}}} u(j) \le u(M) - k\cdot u(j^*),$$

where the second inequality is from the assumption of the lemma. These two inequalities imply the desired contradiction.

Case 2: $j^{\max} \in M_{\text{low-utility}}$. In this case, we must have $u(P_\ell) > w_\ell - u(j^{\max}) \ge w_\ell - u(j^*)/k$ for each $\ell$. Summing this over $\ell \in [k]$, we get

$$u(P_1 \cup \cdots \cup P_k) > w_1 + \cdots + w_k - u(j^*) = u(M) - u(j^*).$$

However, because $j^* \notin P_1 \cup \cdots \cup P_k$, we have $u(P_1 \cup \ldots \cup P_k) \le u(M) - u(j^*)$, which is a contradiction. In both cases, we arrive at a contradiction, and our proof is complete. □

With Lemma 2 ready, we can now prove Theorem 11.

**Proof of Theorem 11.** Because each $u(j)$ is drawn independently from the uniform distribution on $[0,1]$, the probability that $u(j^*) \ge 1/2$ is $1 - 1/2^m$, which converges to one for large $m$. In addition, because $u(j) \in [0.1/k, 0.5/k]$ with probability $0.4/k$ for each $j$, a standard Chernoff bound argument implies that with probability approaching one, we have

$$M' := |\{j \in M \mid u(j) \in [0.1/k, 0.5/k]\}| \ge 0.3m/k.$$

The union bound implies that both events occur simultaneously with high probability. Suppose that they both occur and $m \ge 40k^3$. From the first event, we have $u(j) \le 0.5/k \le u(j^*)/k$ for each $j \in M'$, and so $M' \subseteq M_{\text{low-utility}}$. Hence, the second event implies that

$$\sum_{j \in M_{\text{low-utility}}} u(j) \ge \sum_{j \in M'} u(j) \ge (0.3m/k)(0.1/k) \ge k \ge k \cdot u(j^*).$$

From this and Lemma 2, we conclude that, with high probability, we can efficiently find a consensus $k$-splitting that cuts only a single item as claimed. □

## 7. Conclusion

In this paper, we study a natural version of the consensus halving problem in which, in contrast to prior work, the items do not have a linear ordering. We show that computing a consensus halving with at most $n$ cuts in our version can be done in polynomial time for linear utilities but already becomes PPAD-hard for a class of monotonic utilities that are very close to linear. We also demonstrate several extensions and connections to the problems of consensus $k$-splitting and agreeable sets.

Whereas our PPAD-hardness result serves as strong evidence that consensus halving for a set of items is computationally hard for nonlinear utilities, it remains open whether the result can be strengthened to PPA-completeness; indeed, the membership of the problem in PPA follows from a reduction to consensus halving on a line as explained in the introduction. Obtaining a PPA-hardness result will most likely require new ideas and perhaps even new insights into PPA because all existing PPA-hardness results for consensus halving heavily rely on the linear ordering. Of course, it is also possible that the problem is, in fact, PPAD-complete. In addition to consensus halving, settling the computational complexity of the agreeable sets problem for a nonconstant number of agents with monotonic utilities would also be of interest.

We conclude with further interesting questions that remain from our work.

• Whereas the class of symmetric-threshold utilities that we introduce is a natural generalization of linear utilities, one could also consider other known generalizations. For example, it would be interesting to understand the complexity of consensus halving for separable piecewise linear concave (SPLC) or constant elasticity of substitution (CES) utilities, which are typically considered in economics. Note that symmetric-threshold utilities are separable piecewise linear but usually not concave. Does the concavity of SPLC utilities allow for an extension of our positive results or is the problem also PPAD-hard? For CES utilities, it seems likely that solutions are irrational in general. In that case, the problem of computing an approximate solution remains in PPA, but if one is really interested in exact solutions, then other tools can be used (Batziou et al. [6], Deligkas et al. [16], Etessami and Yannakakis [18]).

• A consensus $k$-splitting always exists for linear utilities because of Theorem 8, and a consensus $1/k$-division always exists even for nonlinear utilities because we can simply divide every item into $k$ equal parts. However, perhaps surprisingly, a consensus $k$-splitting does *not* necessarily exist for nonlinear utilities even for $k = 2$. To see this, suppose that $\alpha_1 = 0.3$ and $\alpha_2 = 0.7$ and there is only one item. If one agent has a linear utility, we have to split the item according to the ratio $0.3 : 0.7$; however, such a split may not satisfy another agent with a nonlinear utility. In light of this observation, what is the "best" approximation to a consensus $k$-splitting that we can provide for nonlinear utilities?

• When $k$ is a prime number, a consensus $1/k$-division using at most $(k-1)n$ cuts exists for the problem on the line even for nonlinear utilities (Filos-Ratsikas et al. [23, theorem 6.5]. As a result, $(k-1)n$ cuts are also sufficient when the items do not lie on a line. From Filos-Ratsikas et al. [23], we also immediately obtain that the corresponding computational problem lies in the class PPA-$k$, a modulo-$k$ analog of PPA (Göös et al. [27], Hollender [30], Papadimitriou [39]). Is it possible to prove a corresponding hardness result, ideally again for simple monotonic utilities? Currently, it is only known that consensus 1/3-division is PPAD-hard on the line (Filos-Ratsikas et al. [22]), but the proof heavily relies on the linear ordering, and it is unclear how to extend this to our setting.

• Can we generalize Theorem 11 to arbitrary $n$, that is, for any fixed $n$ with probability approaching one, as $m \to \infty$, there exists a consensus $k$-splitting with any specified ratios using at most $n$ cuts? We conjecture that this is the case, but it does not seem that our proof of Theorem 11 can be easily extended; indeed, when $n > 1$, in order to use a similar approach, we would have to group the items carefully so that the utilities with respect to these groups are similar for all $n$ agents. More generally, for any fixed $n$, $m$, $k$, and $\alpha_1, \ldots, \alpha_k$, what is the likely minimum number of cuts required in a consensus $k$-splitting with ratios $\alpha_1, \ldots, \alpha_k$?

## Acknowledgments
The authors thank the editors and reviewers of *Mathematics of Operations Research* for their valuable comments. A preliminary version of this paper appeared in the Proceedings of the 16th Conference on Web and Internet Economics (WINE 2020). This version contains a new section with several results for constant numbers of agents (Appendix A) and all proofs (almost all of which were omitted from the conference version).

## Appendix A. Constant Number of Agents
In this section, we provide additional results for the case in which there are a constant number of agents who are endowed with monotonic utilities.

### A.1. Discrete Consensus Halving
We begin by introducing a discrete version of consensus halving, which allows us to focus solely on the agents' utilities for nonfractional sets of items.

**Definition A.1.** A *discrete* consensus halving is a partition of the items into three (nonfractional) sets of items $M_0, M_1, M_2$ such that $u_i(M_0 \cup M_1) \geq u_i(M_2)$ and $u_i(M_0 \cup M_2) \geq u_i(M_1)$ for all $i \in N$.

Note that, for any $r$, a consensus halving with $r$ cuts yields a discrete consensus halving with $|M_0| \leq r$ simply by moving all cut items into $M_0$. Hence, a discrete consensus halving with $|M_0| \leq n$ is guaranteed to exist. The bound $n$ is also tight here: when each agent values a single distinct item, all valued items must be included in $M_0$.

The following result shows that, for constant $n$, a discrete consensus halving with $|M_0| \leq n$ can be found efficiently. Similarly to Theorem 5, the proof involves arranging the items on a line and appealing to the existence of a consensus halving with at most $n$ cuts on the line. As in Theorem 6, we assume a utility oracle model in which the algorithm can query the utility $u_i(M')$ for any $i \in N$ and $M' \subseteq M$.

**Theorem A.1.** *For any constant number of agents with monotonic utilities, there exists a polynomial-time algorithm that computes a discrete consensus halving with* $|M_0| \leq \min\{n, m\}$ *(assuming access to a utility oracle).*

**Proof.** If $n \geq m$, we can simply include all items in $M_0$, so assume that $n \leq m$. Arrange the items on a line in arbitrary order and extend the utility functions of the agents to fractional sets of items in a continuous and monotonic fashion (see Endnote 9). Consider a consensus halving with respect to the extended utilities that uses at most $n$ cuts on the line and move all cut items to $M_0$. The resulting discrete consensus halving has the property that, for any pair of consecutive items in $M_0$, the block of items in between either all belong to $M_1$ or all belong to $M_2$.

We perform a brute-force search over all possible partitions of the items into $M_0, M_1, M_2$ satisfying the preceding property. For $t \in [n]$, there are $O(m^t)$ sets of items that we can choose as $M_0$, and for each choice of $M_0$, there are at most $2^{t+1}$ ways to assign the resulting blocks of items to $M_1$ or $M_2$. Hence, the brute-force search runs in time $\sum_{t=1}^{n} O(2^{t+1} m^t) = O(m^n)$, which is polynomial because $n$ is constant. □

With two agents, the algorithm in Theorem A.1 runs in quadratic time. We next present a more sophisticated algorithm that uses only linear time for this special case. In fact, we show a stronger statement based on a notion introduced by Kyropoulou et al. [31].

**Definition A.2.** Let $n = 2$. A partition of the items into two (nonfractional) sets of items $M_1$ and $M_2$ is said to be *Exact1* if, for each pair $i, k \in \{1, 2\}$, either $M_{3-k} = \emptyset$ or there exists an item $j \in M_{3-k}$ such that $u_i(M_k) \geq u_i(M_{3-k} \setminus \{j\})$.

In words, Exact1 means that, for each agent and each part of the partition, this part can be made at least as valuable as the other part in the agent's view by removing at most one item from the latter part. Given an Exact1 partition, we can easily obtain a discrete consensus halving as follows. From the partition, each agent $i$ proposes (at most) one item to include in $M_0$. Specifically, if $u_i(M_1) < u_i(M_2)$, then agent $i$ proposes an item $j$ such that $u_i(M_1) \geq u_i(M_2 \setminus \{j\})$; the opposite case is analogous. (If $u_i(M_1) = u_i(M_2)$, agent $i$ does not need to propose any item.) It is clear that $|M_0| \leq 2$, and one can check that $(M_0, M_1, M_2)$ forms a discrete consensus halving.

Kyropoulou et al. [31] shows that an Exact1 partition exists for two agents with "responsive utilities," a class that lies between additive and monotonic utilities. Here, we present an algorithm that computes an Exact1 partition for arbitrary monotonic utilities in linear time; to the best of our knowledge, even the *existence* of such a partition has not been established before.

Our algorithm is based on carefully discretizing a procedure of Austin [4], which computes a (nondiscrete) consensus halving for two agents assuming that the resource is represented by the circumference of a circle. Austin's procedure works by letting the first agent place two knives on the circle so that the item is cut in half according to the agent's valuation. The agent then moves both knives continuously clockwise, maintaining the invariant that the knives divide the items into two equal halves in the agent's opinion. The first agent stops moving the knives when the two parts are equal according to the valuation of the second agent, and the procedure returns the resulting partition. Because the second knife reaches the initial position of the first knife at the same time as the first knife reaches the starting point of the second knife, it follows from the intermediate value theorem that the procedure necessarily terminates.

The main challenge in applying this procedure to our discrete item setting is that it is not a priori clear how to implement moving both knives simultaneously; indeed, moving each of the knives over one item does not always maintain the invariant that the partition is Exact1. Nevertheless, as we show, this invariant can be maintained by either moving both knives or moving one of the two knives, whichever option is appropriate at each stage. In fact, for this algorithm and proof, we use a slightly stronger definition of Exact1 wherein the items lie on a circle, each part of the partition forms a contiguous block on the circle, and the item $j$ in Definition A.2 is only allowed to be one of the items at the end of block $M_{3-k}$.[11] We say that a partition is Exact1 for agent $i$ if the (stronger) Exact1 condition is fulfilled for agent $i$ and both $k \in \{1,2\}$.

**Algorithm A.1** (For Two Agents with Monotonic Utilities)

Step 1: Arrange the items on a circle in arbitrary order. Place the first knife between two arbitrary consecutive items on the circle and the second knife between two items so that the partition induced by the two knives is Exact1 for the first agent.

Step 2: If the current partition is Exact1 for the second agent, return this partition.

Step 3: If one of the knives is at the initial position of the other knife, go to step 4. Else, perform one of the following actions so that the new partition remains Exact1 for the first agent:

a. Move the first knife clockwise by one position.
b. Move the second knife clockwise by one position.
c. Move each of the two knives clockwise by one position.

Go back to step 2.

Step 4: Move the knife that is not at the initial position of the other knife clockwise by one position. Go back to step 2.

**Theorem A.2.** *For two agents with monotonic utilities, Algorithm A.1 computes an Exact1 partition in time linear in m (assuming access to a utility oracle).*

**Proof.** Observe that, throughout the algorithm, the partition induced by the two knives is Exact1 for the first agent. Moreover, a partition is returned only if it is Exact1 for the second agent. Hence, if the algorithm terminates, the partition that it outputs is Exact1 for both agents. It, therefore, suffices to establish that the algorithm is well-defined and always terminates. For convenience, we say that a bundle is envy-free up to one item (EF1) for a specific agent if the Exact1 condition (specifically, the stronger version described before the algorithm) is fulfilled for the agent when that bundle is taken as $M_k$.

First, we need to show that in step 1, there exists a position of the second knife such that the resulting partition is Exact1 for the first agent. It turns out that this already follows from Oh et al. [38, theorem 3.1], so the first step can be implemented.

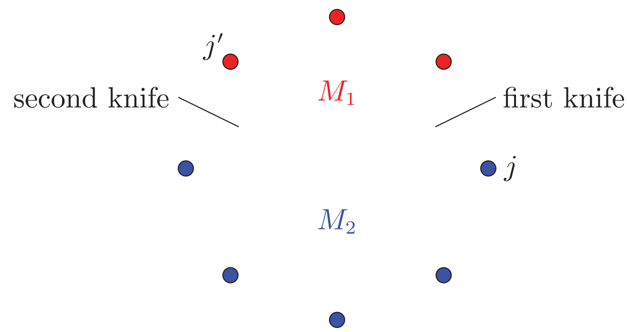
Next, the key part of our proof is to show that, in step 3, at least one of the three actions keeps the new partition Exact1 for the first agent. Assume that actions (a) and (b) do not; we claim that action (c) does. Call the two parts of the partition $M_1$ and $M_2$ and assume without loss of generality that moving the first knife clockwise enlarges $M_1$. Suppose that the next item that the first knife moves over is $j$, and the next item that the second knife moves over is $j'$. (See Figure A.1 for an illustration.) Let $O_1 = (M_1 \cup \{j\}) \backslash \{j'\}$ and $O_2 = (M_2 \cup \{j'\}) \backslash \{j\}$ be the two parts of the partition that results from action (c). Because action (a) does not keep the partition Exact1, we have that $M_2 \backslash \{j\}$ is not EF1 for the first agent. Hence,

$$u_1(O_1) = u_1((M_1 \cup \{j\}) \backslash \{j'\}) > u_1(M_2 \backslash \{j\}) = u_1(O_2 \backslash \{j'\}),$$

implying that $O_1$ is EF1 for the first agent. By symmetry, because action (b) does not keep the partition Exact1, we have that $M_1 \backslash \{j'\}$ is not EF1 for the first agent. This implies that $O_2$ is EF1 for the agent. It follows that action (c) keeps the partition Exact1 for the first agent as claimed.

Now, consider step 4. Because each knife never moves by more than one position at a time, unless the algorithm terminates beforehand, this step eventually is reached. Suppose that the first knife has arrived at the initial position of the second knife, but the second knife is not yet at the initial position of the first knife. The current partition is Exact1 for the first agent. Also, if the second knife moves clockwise to the initial position of the first knife, again we have an Exact1

**Figure A.1.** An illustration of Algorithm A.1.




partition for the agent. Hence, monotonicity of the EF1 property implies that every partition in between is also Exact1 for the agent.

Finally, we show that the algorithm necessarily terminates. Suppose that this is not the case. Assume that, in the initial partition with parts $M_1$ and $M_2$, the second agent believes that $M_2$ is not EF1. This means that $u_2(M_2) < u_2(M_1 \backslash \{j\})$ for any $j$ at the end of block $M_1$. In one iteration of step 3 or 4, $M_1$ loses at most one end item to $M_2$—call this item $j'$ (if $M_1$ does not lose any item, take $j'$ to be an arbitrary end item in $M_1$)—and the respective parts of the partition after the iteration $O_1$ and $O_2$. Because $u_2(M_1 \backslash \{j'\}) > u_2(M_2) = u_2((M_2 \cup \{j'\}) \backslash \{j'\})$, we have that $O_1$ is also EF1 for the second agent. However, because the algorithm does not terminate here by assumption, $O_2$ is not EF1 for the second agent. The same argument tells us that, in further iterations, the second bundle (i.e., $M_2$, $O_2$, and so on) is still not EF1 for the agent. However, the algorithm must reach a point at which the first knife is at the initial position of the second knife and, at the same time, the second knife is also at the initial position of the first knife. At this point, the second bundle coincides with the initial first bundle, so it must be EF1 for the second agent. This yields the desired contradiction.

Regarding the running time, note that each knife moves clockwise around the circle only once, so the number of partitions considered by the algorithm is linear. For each partition, checking the relevant Exact1 condition can be done in constant time because it involves hypothetically removing only a constant number of items. Hence, the algorithm runs in linear time as claimed. □

### A.2. Continuous Extensions

The discrete consensus halving problem allows us to concern ourselves exclusively with the agents' utilities for nonfractional sets of items, which are represented by set functions. For an additive set function, there exists an obvious extension to fractional sets of items: the linear extension used in Section 2. This is, however, not the case for general monotonic functions. In this section, we address two extensions that are studied in the literature, namely, the Lovász and multilinear extensions. We refer to the lecture notes of Vondrák [54] for further discussion of these extensions.

Let $\mathbf{x} = (x_1, \ldots, x_m)$, and for each subset $S \subseteq [m]$, denote by $\mathbf{1}_S$ the vector of length $m$ such that the $i$th component is one if $i \in S$ and zero otherwise.

**Definition A.3.** Given a function $f : \{0,1\}^m \to \mathbb{R}$, the *Lovász extension* $f^L : [0,1]^m \to \mathbb{R}$ of $f$ is defined by

$$f^L(\mathbf{x}) = \sum_{i=0}^{m} \lambda_i f(S_i),$$

where $\emptyset = S_0 \subset S_1 \subset \cdots \subset S_m = [m]$ is a chain such that $\sum_{i=0}^{m} \lambda_i \mathbf{1}_{S_i} = \mathbf{x}$ for $\lambda_0, \lambda_1, \ldots, \lambda_m \geq 0$ with $\sum_{i=0}^{m} \lambda_i = 1$.

As an example, suppose that $m = 3$ and $\mathbf{x} = (1, 0.1, 0.3)$. Then, we have $S_1 = \{1\}$, $S_2 = \{1,3\}$, $S_3 = \{1,2,3\}$, and $(\lambda_0, \lambda_1, \lambda_2, \lambda_3) = (0, 0.7, 0.2, 0.1)$, meaning that

$$f^L(\mathbf{x}) = 0.7 \cdot f(\{1\}) + 0.2 \cdot f(\{1,3\}) + 0.1 \cdot f(\{1,2,3\}).$$

**Definition A.4.** Given a function $f : \{0,1\}^m \to \mathbb{R}$, the *multilinear extension* $F : [0,1]^m \to \mathbb{R}$ of $f$ is defined by

$$F(\mathbf{x}) = \sum_{S \subseteq [m]} f(S) \prod_{i \in S} x_i \prod_{i \in [m] \backslash S} (1 - x_i).$$

For this example, we have

$$\begin{aligned} F(\mathbf{x}) &= 0.9 \cdot 0.7 \cdot f(\{1\}) + 0.1 \cdot 0.7 \cdot f(\{1,2\}) + 0.9 \cdot 0.3 \cdot f(\{1,3\}) + 0.1 \cdot 0.3 \cdot f(\{1,2,3\}) \\ &= 0.63 \cdot f(\{1\}) + 0.07 \cdot f(\{1,2\}) + 0.27 \cdot f(\{1,3\}) + 0.03 \cdot f(\{1,2,3\}). \end{aligned}$$

Vondrák [54] proves that, if $f$ is a monotonic set function, then its multilinear extension $F$ is also monotonic; that is, increasing a component $x_i$ by any amount does not decrease the value of the function $F(\mathbf{x})$. For completeness, we show an analogous result for the Lovász extension.

**Proposition A.1.** *If a function $f:\{0,1\}^m \to \mathbb{R}$ is monotonic, then so is its Lovász extension $f^L$.*

**Proof.** Let $f$ be a monotonic set function and $f^L$ be its Lovász extension. Let $\mathbf{x} \in [0,1]^m$ and assume that $x_1 \le x_2 \le \cdots \le x_m$ (other orderings can be handled analogously). In this case, we have

$$\begin{aligned} f^L(\mathbf{x}) = x_1 f(\{1,2,\ldots,m\}) + (x_2 - x_1) f(\{2,3,\ldots,m\}) + \ldots \\ + (x_i - x_{i-1}) f(\{i,\ldots,m\}) + (x_{i+1} - x_i) f(\{i+1,\ldots,m\}) + \ldots \\ + (x_m - x_{m-1}) f(\{m\}). \end{aligned}$$

It suffices to show that, for any $i$, the value $f^L(\mathbf{x})$ does not decrease upon increasing $x_i$. This is obvious if $i = m$. For $1 \le i \le m-1$, we only need to prove that $f^L(\mathbf{x})$ does not decrease when we increase $x_i$ until it reaches $x_{i+1}$; indeed, if we want to increase $x_i$ further, we can swap the roles of $x_i$ and $x_{i+1}$ and apply the same argument. When we increase $x_i$ in the range $[x_{i-1}, x_{i+1}]$, the only terms that change are $x_i \cdot f(\{i,\ldots,m\})$ and $-x_i \cdot f(\{i+1,\ldots,m\})$. The net change is

$$x_i \cdot (f(\{i,\ldots,m\}) - f(\{i+1,\ldots,m\})),$$

which is nonnegative because of the monotonicity of $f$. The conclusion follows. □

When $n$ is constant, computing a consensus halving for a utility function given by the Lovász extension of a monotonic set function can be done efficiently.

**Theorem A.3.** *For a constant number of agents with monotonic utilities, each given by the Lovász extension of a set function, there exists a polynomial-time algorithm that computes a consensus halving with at most* $\min\{n,m\}$ *cuts (assuming access to a utility oracle for the set function).*

**Proof.** If $n \ge m$, we can simply divide every item in half, so assume that $n \le m$. Arrange the items on a line in arbitrary order. Similarly to the proof of Theorem A.1, there exists a consensus halving that uses at most $n$ cuts on the line such that, for any pair of consecutive cut items, the block of whole items in between all belong to either $M_1$ or $M_2$. We perform a brute-force search over all partitions of items into $(M_0, M_1, M_2)$ such that all cut items belong to $M_0$ and the property is satisfied; as in Theorem A.1, this search takes polynomial time.

For each such partition, it remains to determine the ratios by which we should divide the items in $M_0$ between $M_1$ and $M_2$. Denote by $x_1,\ldots,x_r$ the fraction of the $r \le n$ items in $M_0$ that should go into $M_1$. We iterate over all possible orderings of $x_1,\ldots,x_r$; there are at most $n!$ orderings, which is polynomial because $n$ is constant. For each ordering, one can verify that the consensus halving condition for each agent reduces to a linear equation in $x_1,\ldots,x_r$. Hence, to check the feasibility of a partition along with an ordering, we can run any efficient linear programming algorithm (with an arbitrary objective) on the ordering and consensus halving constraints. The previous paragraph implies that at least one combination of partition and ordering results in a feasible linear program, which, in turn, gives rise to the desired consensus halving. □

A consequence of Theorem A.3 is that, for the Lovász extension, if the set function is rational, then there exists a consensus halving with rational ratios. By contrast, for the multilinear extension, a consensus halving may necessarily involve splitting items in irrational ratios even if the set function only takes on integer values.

**Theorem A.4.** *There exists an instance with $n = 2$ and $m = 3$ in which each agent has a monotonic utility function given by the multilinear extension of a set function taking on integer values, but every consensus halving with at most two cuts involves splitting some items in irrational ratios.*

**Proof.** Assume that $n = 2$ and $m = 3$. The utility functions of the agents are given in Table A.1. Notice that the function of the second agent is the same as that of the first agent except with the roles of items 2 and 3 reversed.

Consider a consensus halving $(M_1, M_2)$ of this instance with at most two cuts. Because $u_1(2) > u_1(\{1,3\})$, item 2 needs to be cut. Similarly, because $u_2(3) > u_2(\{1,2\})$, item 3 needs to be cut. Hence, item 1 must be uncut; assume without loss of generality that it belongs to $M_1$. Let $x_2$ and $x_3$ be the fractions of item 2 and 3 in $M_1$, respectively. Because $u_1(M_1) = u_1(M_2)$, we have

$$\begin{aligned} &(1-x_2)(1-x_3)\cdot u_1(1) + x_2(1-x_3)\cdot u_1(\{1,2\}) + x_3(1-x_2)\cdot u_1(\{1,3\}) + x_2x_3\cdot u_1(\{1,2,3\}) \\ &= x_2x_3 \cdot u_1(\emptyset) + x_3(1-x_2)\cdot u_1(2) + x_2(1-x_3)\cdot u_1(3) + (1-x_2)(1-x_3)\cdot u_1(\{2,3\}). \end{aligned}$$

This is equivalent to

$$(1-x_2)(1-x_3)\cdot(-13) + x_2(1-x_3)\cdot 10 + x_3(1-x_2)\cdot(-7) + x_2x_3 \cdot 20 = 0,$$

or

$$-13 + 23x_2 + 6x_3 + 4x_2x_3 = 0. \quad \text{(A.1)}$$

By symmetry, $u_2(M_1) = u_2(M_2)$ implies that

$$-13 + 23x_3 + 6x_2 + 4x_2x_3 = 0. \quad \text{(A.2)}$$

**Table A.1.** Utility functions for the instance in the proof of Theorem A.4.

| Set $S$ | $u_1(S)$ | $u_2(S)$ |
|---|---|---|
| $\emptyset$ | 0 | 0 |
| {1} | 1 | 1 |
| {2} | 10 | 2 |
| {3} | 2 | 10 |
| {1, 2} | 12 | 3 |
| {1, 3} | 3 | 12 |
| {2, 3} | 14 | 14 |
| {1, 2, 3} | 20 | 20 |

Subtracting (A.2) from (A.1) yields $17x_2 = 17x_3$, so $x_2 = x_3$. Plugging this back into (A.1), we get

$$4x_2^2 + 29x_2 - 13 = 0. \tag{A.3}$$

The only positive solution to (A.3) is $x_2 = \frac{\sqrt{1049}-29}{8} \approx 0.4235\ldots$, meaning that every consensus halving involves splitting items 2 and 3 in irrational ratios. □

Theorem A.4 implies that, for the multilinear extension, computing a consensus halving exactly may not be possible if our computation model only allows representing rational numbers. As we can see, with two agents and two necessary cuts, the problem already requires solving a quadratic equation. For more agents, we can, therefore, expect that one would need to solve higher degree polynomial equations; the Abel–Ruffini theorem states that almost all polynomials of degree at least five do not admit a solution in radicals. Hence, for this extension, finding an approximate consensus halving is likely the best that one could do even under general computational models.

## Appendix B. Proof of Lemma 1

We reduce from the $\varepsilon$-GCIRCUIT problem, which is known to be PPAD-hard even for some constant $\varepsilon > 0$ (Rubinstein [42]). In this problem, we are given a generalized circuit $(V, \mathcal{T})$, where there are nine gate types: $G_\zeta$, $G_{\times\zeta}$, $G_=$, $G_+$, $G_-$, $G_<$, $G_\vee$, $G_\wedge$, and $G_\neg$ with $\zeta \in [0,1]$ for the first two gates (see Rubinstein [42] for a formal definition of the gates). The last three gate types correspond to Boolean operations. As shown by Schuldenzucker and Seuken [44, corollary 1], these three gate types are actually not necessary, and the problem remains PPAD-hard for constant $\varepsilon$ even without them. Apart from the set of gates, the other difference with $\varepsilon$-SIMPLE-GCIRCUIT is that, in $\varepsilon$-GCIRCUIT, we want to assign a number in $[0,1]$ to each node (instead of $[-1,1]$).

Let $\widehat{\varepsilon} > 0$ be a constant such that the $\widehat{\varepsilon}$-GCIRCUIT problem without Boolean operation gates is PPAD-hard and let $(V, \mathcal{T})$ be an instance of $\widehat{\varepsilon}$-GCIRCUIT without Boolean gates. We construct an instance $(V', \mathcal{T}')$ of $\varepsilon$-SIMPLE-GCIRCUIT, in which $\varepsilon > 0$ is a sufficiently small constant (which we pick later) such that any solution to the new instance yields a solution to the original instance. We let $V' = V \cup V_{aux}$, where $V_{aux}$ is a set of nodes that is used for "intermediate" results when simulating the gates of the original problem with the restricted set of gates allowed in $\varepsilon$-SIMPLE-GCIRCUIT. We construct $\mathcal{T}'$ such that it induces the original constraints of $\mathcal{T}$ on the nodes $V \subset V'$. Furthermore, we also ensure that, in any solution $\mathbf{x} : V' \to [-1,1]$, we have $\mathbf{x}[v] \in [0,1]$ for all $v \in V \subset V'$. Thus, restricting $\mathbf{x}$ to $V$ immediately yields a solution to the original $\widehat{\varepsilon}$-GCIRCUIT instance.

Recall that we only have three types of gates at our disposal: $G_+$, $G_{\times-\zeta}$ for $\zeta \in (0,1]$, and $G_1$. We begin by constructing some useful gadgets that simulate more operations on the same domain $[-1,1]$. Throughout, we denote the input nodes by $u_1$, $u_2$ (if applicable) and the output node by $v$.

### B.1. $G_{\times\zeta}$: Multiplication by $\zeta \in [-1,1]$

This gadget ensures that $\mathbf{x}[v] = \zeta \cdot \mathbf{x}[u_1] \pm 2\varepsilon$. If $\zeta < 0$, then $-\zeta \in (0,1]$, and we can simply use a $G_{\times-(-\zeta)}$ gate with input $u_1$ and output $v$. If $\zeta > 0$, we use a $G_{\times-\zeta}$ gate with input $u_1$ and output $w \in V_{aux}$ and then a $G_{\times-1}$ gate with input $w$ and output $v$, which ensures that $\mathbf{x}[v] = \zeta \cdot \mathbf{x}[u_1] \pm 2\varepsilon$. Finally, if $\zeta = 0$, then we use a $G_{\times-1}$ gate with input $u_1$ and output $w \in V_{aux}$ and then a $G_+$ gate with inputs $u_1, w$ and output $v$. This ensures that $\mathbf{x}[v] = 0 \pm 2\varepsilon$.

### B.2. $G_\zeta$: Constant $\zeta \in [-1,1]$

This gadget ensures that $\mathbf{x}[v] = \zeta \pm 3\varepsilon$. We use a $G_1$ gate with output $w \in V_{aux}$ and then a $G_{\times\zeta}$ gadget with input $w$ and output $v$, which yields the desired result.

### B.3. $G_{\times 2}$: Multiplication by Two

This gadget ensures that $\mathbf{x}[v] = T_{[-1,1]}(2\mathbf{x}[u_1]) \pm 3\varepsilon$. We use a $G_{\times 1}$ gadget with input $u_1$ and output $w \in V_{aux}$ and then a $G_+$ gate with inputs $u_1, w$ and output $v$, which yields the desired result.

Before we show how to construct gadgets that simulate the gates of $\widehat{\varepsilon}$-GCIRCUIT, we need a way to ensure that, for $v \in V \subset V'$, we have $\mathbf{x}[v] \in [0,1]$. To achieve this, we make extensive use of the following gadget.

### B.4. $G_{[0,1]}$: Truncation to $[0,1]$

This gadget ensures that $\mathbf{x}[v] \in [0,1]$ and $\mathbf{x}[v] = T_{[0,1]}(\mathbf{x}[u_1]) \pm 16\varepsilon$. To achieve this, we use the fact that, for any $t \in [-1,1]$, it holds that $T_{[0,1]}(t) = T_{[-1,1]}[t + (-1)] + 1$. First, we use a $G_{-1}$ gadget with output $w_1 \in V_{aux}$, and then a $G_+$ gate with inputs $u_1$, $w_1$ and output $w_2 \in V_{aux}$. Next, we use a $G_1$ gate with output $w_3 \in V_{aux}$ and then a $G_+$ gate with inputs $w_2$, $w_3$ and output $w_4 \in V_{aux}$. Because the $G_{-1}$ gadget has error at most $3\varepsilon$ and the $G_+$ and $G_1$ gates have error at most $\varepsilon$, we obtain that $\mathbf{x}[w_4] = T_{[0,1]}(\mathbf{x}[u_1]) \pm 6\varepsilon$. Furthermore, it holds that $\mathbf{x}[w_4] \geq -2\varepsilon$ because $\mathbf{x}[w_4] = T_{[-1,1]}(\mathbf{x}[w_2] + \mathbf{x}[w_3]) \pm \varepsilon$, $\mathbf{x}[w_2] \in [-1,1]$ and $\mathbf{x}[w_3] \geq 1 - \varepsilon$. Finally, we also use a $G_{6\varepsilon}$ gadget with output $w_5 \in V_{aux}$ and a $G_+$ gate with inputs $w_4$, $w_5$ and output $v$. This introduces an additional error of at most $4\varepsilon$ and, thus, ensures that $\mathbf{x}[v] = T_{[-1,1]}(T_{[0,1]}(\mathbf{x}[u_1]) + 6\varepsilon) \pm 10\varepsilon = T_{[0,1]}(\mathbf{x}[u_1]) \pm 16\varepsilon$. Furthermore, it also holds that $\mathbf{x}[v] \geq T_{[-1,1]}(\mathbf{x}[w_4] + \mathbf{x}[w_5]) - \varepsilon \geq 0$ because $\mathbf{x}[w_4] \geq -2\varepsilon$ and $\mathbf{x}[w_5] \geq 6\varepsilon - 3\varepsilon$.

We are now ready to simulate the constraints $\mathcal{T}$ of the original instance on the nodes $V \subset V'$. First of all, for any node $v \in V$ that does not appear as the output of any gate in $\mathcal{T}$, we ensure that $\mathbf{x}[v] \in [0,1]$ as follows: create a node $w \in V_{aux}$ and use a $G_{[0,1]}$ gadget with input $w$ and output $v$. Note that we do not care about the error in this case because we only want to ensure that $\mathbf{x}[v] \in [0,1]$. For all $v \in V$ that appear as the output of some gate in $\mathcal{T}$, the gadget that outputs into $v$ ensures that $\mathbf{x}[v] \in [0,1]$.

For every gate $T = (G, u_1, u_2, v, \zeta) \in \mathcal{T}$, we ensure that the corresponding constraint holds as follows.

### B.5. $(G_{\zeta}^{[0,1]}, nil, nil, v, \zeta)$: Constant $\zeta \in [0,1]$

This gadget ensures that $\mathbf{x}[v] \in [0,1]$ and $\mathbf{x}[v] = \zeta \pm 19\varepsilon$. We use a $G_\zeta$ gadget with output $w \in V_{aux}$ and then a $G_{[0,1]}$ gadget with input $w$ and output $v$.

### B.6. $(G_{\times\zeta}^{[0,1]}, u_1, nil, v, \zeta)$: Multiplication by $\zeta \in [0,1]$

This gadget ensures that $\mathbf{x}[v] \in [0,1]$ and $\mathbf{x}[v] = T_{[0,1]}(\zeta \cdot \mathbf{x}[u_1]) \pm 18\varepsilon$. We use a $G_{\times\zeta}$ gadget with input $u_1$ and output $w \in V_{aux}$ and then a $G_{[0,1]}$ gadget with input $w$ and output $v$.

### B.7. $(G_{=}^{[0,1]}, u_1, nil, v, nil)$: Copy

This gadget ensures that $\mathbf{x}[v] \in [0,1]$ and $\mathbf{x}[v] = T_{[0,1]}(\mathbf{x}[u_1]) \pm 16\varepsilon$. For this, we simply use the $G_{[0,1]}$ gadget with input $u_1$ and output $v$.

### B.8. $(G_{+}^{[0,1]}, u_1, u_2, v, nil)$: Addition

This gadget ensures that $\mathbf{x}[v] \in [0,1]$ and $\mathbf{x}[v] = T_{[0,1]}(\mathbf{x}[u_1] + \mathbf{x}[u_2]) \pm 17\varepsilon$. We use a $G_+$ gate with inputs $u_1$, $u_2$ and output $w \in V_{aux}$ and then a $G_{[0,1]}$ gadget with input $w$ and output $v$.

### B.9. $(G_{-}^{[0,1]}, u_1, u_2, v, nil)$: Subtraction

This gadget ensures that $\mathbf{x}[v] \in [0,1]$ and $\mathbf{x}[v] = T_{[0,1]}(\mathbf{x}[u_1] - \mathbf{x}[u_2]) \pm 18\varepsilon$. We use a $G_{\times-1}$ gate with input $u_2$ and output $w_1 \in V_{aux}$ and then a $G_+$ gate with inputs $u_1$, $w_1$ and output $w_2 \in V_{aux}$ and, finally, a $G_{[0,1]}$ gadget with input $w_2$ and output $v$.

### B.10. $(G_{<}^{[0,1]}, u_1, u_2, v, nil)$: Comparison

This gadget ensures that $\mathbf{x}[v] \in [0,1]$ and

- If $\mathbf{x}[u_1] < \mathbf{x}[u_2] - \widehat{\varepsilon}$, then $\mathbf{x}[v] = 1 \pm 19\varepsilon$.
- If $\mathbf{x}[u_1] > \mathbf{x}[u_2] + \widehat{\varepsilon}$, then $\mathbf{x}[v] = 0 \pm 19\varepsilon$.

We use a $G_{\times-1}$ gate with input $u_1$ and output $w \in V_{aux}$ and then a $G_+$ gate with inputs $u_2, w$ and output $w_0 \in V_{aux}$. This ensures that $\mathbf{x}[w_0] = T_{[-1,1]}(\mathbf{x}[u_2] - \mathbf{x}[u_1]) \pm 2\varepsilon$. Let $k = \lceil \log_2(1/\widehat{\varepsilon}) \rceil + 1$, so $2/\widehat{\varepsilon} \leq 2^k \leq 4/\widehat{\varepsilon}$. Next, for $i \in [k]$, we use a $G_{\times 2}$ gadget with input $w_{i-1}$ and output $w_i \in V_{aux}$. Finally, we use a $G_{[0,1]}$ gadget with input $w_k$ and output $v$.

For the analysis, we first consider the case $\mathbf{x}[u_1] < \mathbf{x}[u_2] - \widehat{\varepsilon}$. Then, it holds that $\mathbf{x}[w_0] \geq \widehat{\varepsilon} - 2\varepsilon \geq \widehat{\varepsilon} - 3\varepsilon$. First, let us show by contradiction that there must exist $i \in [k]$ such that $\mathbf{x}[w_i] < 2\mathbf{x}[w_{i-1}] - 3\varepsilon$. Assume that, for all $i \in [k]$, we have $\mathbf{x}[w_i] \geq 2\mathbf{x}[w_{i-1}] - 3\varepsilon$. Then, it follows that $\mathbf{x}[w_k] \geq 2^k \widehat{\varepsilon} - 3\varepsilon \sum_{i=0}^{k} 2^i \geq 2^k \widehat{\varepsilon} - 3\varepsilon 2^{k+1} \geq 2\widehat{\varepsilon}/\widehat{\varepsilon} - 3\varepsilon \cdot 8/\widehat{\varepsilon} = 2 - 24\varepsilon/\widehat{\varepsilon}$. If we ensure that $\varepsilon < \widehat{\varepsilon}/24$, then we obtain that $\mathbf{x}[w_k] > 1$, which is impossible. Thus, let $i \in [k]$ be such that $\mathbf{x}[w_i] < 2\mathbf{x}[w_{i-1}] - 3\varepsilon$. Recall that the $G_{\times 2}$ gadget ensures that $\mathbf{x}[w_i] \geq T_{[-1,1]}(2\mathbf{x}[w_{i-1}]) - 3\varepsilon$. Thus, it must be that $T_{[-1,1]}(2\mathbf{x}[w_{i-1}]) < 2\mathbf{x}[w_{i-1}]$, that is, $2\mathbf{x}[w_{i-1}] \geq 1$. This implies that $\mathbf{x}[w_i] \geq 1 - 3\varepsilon$, and thus, $\mathbf{x}[w_{i+1}] \geq T_{[-1,1]}(2 - 6\varepsilon) - 3\varepsilon \geq 1 - 3\varepsilon$ if we ensure that $\varepsilon \leq 1/6$. By induction, it follows that $\mathbf{x}[w_k] \geq 1 - 3\varepsilon$, and thus, $\mathbf{x}[v] \geq 1 - 19\varepsilon$ as desired.

Now, consider the case $\mathbf{x}[u_1] > \mathbf{x}[u_2] + \widehat{\varepsilon}$. By an analogous argument, we obtain that it must be that $\mathbf{x}[w_k] \leq -1 + 3\varepsilon$, and thus, $\mathbf{x}[v] = T_{[0,1]}(\mathbf{x}[w_k]) \pm 16\varepsilon = 0 \pm 19\varepsilon$.

We can now finish the reduction. We set $\varepsilon = \widehat{\varepsilon}/25$. This ensures that all the assumptions we have made about $\varepsilon$ hold and that all the gadgets that simulate the gates in $\mathcal{T}$ have error at most $\widehat{\varepsilon}$.

## Endnotes

[1] Simmons and Su [50] assume that the resource is a two- or three-dimensional object but only consider cuts by parallel planes; their model is, therefore, equivalent to that of a one-dimensional object.

[2] These classes were introduced by Papadimitriou [39]. These phrases refer to the combinatorial principles that guarantee the existence of solutions. For details, we refer to Roughgarden [41, chapter 20].

[3] This version, together with some further simplifications, is used in subsequent work by Filos-Ratsikas et al. [24] to prove that computing an approximate equilibrium of a first-price auction with subjective priors is PPAD-hard.

[4] See http://www.nyu.edu/projects/adjustedwinner for a demonstration and implementation of the procedure.

[5] This motivates relaxations such as envy-freeness up to one item (EF1) and envy-freeness up to any item (EFX), which have been extensively studied in the last few years (e.g., Caragiannis et al. [11], Plaut and Roughgarden [40]). However, as Sandomirskiy and Segal-Halevi [43] note, when a divorcing couple decides how to split their children or two siblings try to divide three houses between them, it is unlikely that anyone will agree to a bundle that is envy-free up to one child or house.

[6] Specifically, if the linear equations in $S \cup T$ lead to a unique solution $(x_1, \ldots, x_m)$, then Gaussian elimination immediately results in this solution. Otherwise, Gaussian elimination yields a row echelon form; by setting one of the nonpivots $y_j$ to be an arbitrary number not equal to $x_j$, we obtain a solution that is not equal to $(x_1, \ldots, x_m)$.

[7] This means that, for each agent $i$, the interval $[0, 1]$ can be partitioned into a finite number of intervals so that the density of the agent's valuation function is either zero or some constant $c_i$ over each interval.

[8] The notion of agreeability was introduced in an earlier conference version of the paper (Suksompong [52]). Gourvès [28] considered an extension of the problem that takes into account matroidal constraints.

[9] For example, one can use the Lovász or multilinear extension (see Section A.2).

[10] See the definition of the consensus halving problem on a line before Theorem 2.

[11] A notion in the same spirit called "envy-freeness up to one outer good" is proposed by Bilò et al. [7].

## References

[1] Alon N (1987) Splitting necklaces. *Adv. Math.* 63(3):247–253.
[2] Alon N, Graur A (2021) Efficient splitting of necklaces. *Proc. 48th Internat. Colloquium Automata, Languages, Programming*, 1–17.
[3] Alon N, West DB (1986) The Borsuk-Ulam theorem and bisection of necklaces. *Proc. Amer. Math. Soc.* 98(4):623–628.
[4] Austin AK (1982) Sharing a cake. *Math. Gazette* 66(437):212–215.
[5] Aziz H, Caragiannis I, Igarashi A, Walsh T (2019) Fair allocation of indivisible goods and chores. *Proc. 28th Internat. Joint Conf. Artificial Intelligence*, 53–59.
[6] Batziou E, Hansen KA, Høgh K (2021) Strong approximate consensus halving and the Borsuk-Ulam theorem. *Proc. 48th Internat. Colloquium Automata, Languages, Programming*, 1–20.
[7] Bilò V, Caragiannis I, Flammini M, Igarashi A, Monaco G, Peters D, Vinci C, Zwicker WS (2019) Almost envy-free allocations with connected bundles. *Proc. 10th Innovations Theoretical Comput. Sci. Conf.*, 1–21.
[8] Bogomolnaia A, Moulin H, Sandomirskiy F, Yanovskaya E (2017) Competitive division of a mixed manna. *Econometrica* 85(6):1847–1871.
[9] Brams SJ, Taylor AD (1996) *Fair Division: From Cake-Cutting to Dispute Resolution* (Cambridge University Press).
[10] Brams SJ, Taylor AD (1999) *The Win-Win Solution: Guaranteeing Fair Shares to Everybody* (W. W. Norton & Company).
[11] Caragiannis I, Kurokawa D, Moulin H, Procaccia AD, Shah N, Wang J (2019) The unreasonable fairness of maximum Nash welfare. *ACM Trans. Econom. Comput.* 7(3):1–32.
[12] Chaudhury BR, Garg J, McGlaughlin P, Mehta R (2021) Competitive allocation of a mixed manna. *Proc. 32nd ACM-SIAM Sympos. Discrete Algorithms*, 1405–1424.
[13] Chen X, Deng X, Teng SH (2009) Settling the complexity of computing two-player Nash equilibria. *J. ACM* 56(3):1–57.
[14] Daskalakis C, Goldberg PW, Papadimitriou CH (2009) The complexity of computing a Nash equilibrium. *SIAM J. Comput.* 39(1):195–259.
[15] Deligkas A, Filos-Ratsikas A, Hollender A (2021) Two's company, three's a crowd: Consensus-halving for a constant number of agents. *Proc. 22nd ACM Conf. Econom. Comput.*, 347–368.
[16] Deligkas A, Fearnley J, Melissourgos T, Spirakis PG (2021) Computing exact solutions of consensus halving and the Borsuk-Ulam theorem. *J. Comput. System Sci.* 117:75–98.
[17] Deng X, Qi Q, Saberi A (2012) Algorithmic solutions for envy-free cake cutting. *Oper. Res.* 60(6):1461–1476.
[18] Etessami K, Yannakakis M (2010) On the complexity of Nash equilibria and other fixed points. *SIAM J. Comput.* 39(6):2531–2597.
[19] Filos-Ratsikas A, Goldberg PW (2018) Consensus halving is PPA-complete. *Proc. 50th Annual ACM SIGACT Sympos. Theory Comput.*, 51–64.
[20] Filos-Ratsikas A, Goldberg PW (2019) The complexity of splitting necklaces and bisecting ham sandwiches. *Proc. 51st Annual ACM SIGACT Sympos. Theory Comput.*, 638–649.
[21] Filos-Ratsikas A, Frederiksen SKS, Goldberg PW, Zhang J (2018) Hardness results for consensus-halving. *Proc. 43rd Internat. Sympos. Math. Foundations Comput. Sci.*, 1–16.
[22] Filos-Ratsikas A, Hollender A, Sotiraki K, Zampetakis M (2020) Consensus-halving: Does it ever get easier? *Proc. 21st ACM Conf. Econom. Comput.*, 381–399.
[23] Filos-Ratsikas A, Hollender A, Sotiraki K, Zampetakis M (2021) A topological characterization of modulo-$p$ arguments and implications for necklace splitting. *Proc. 32nd ACM-SIAM Sympos. Discrete Algorithms*, 2615–2634.
[24] Filos-Ratsikas A, Giannakopoulos Y, Hollender A, Lazos P, Poças D (2021) On the complexity of equilibrium computation in first-price auctions. *Proc. 22nd ACM Conf. Econom. Comput.*, 454–476.
[25] Garg J, Mehta R, Vazirani VV (2018) Substitution with satiation: A new class of utility functions and a complementary pivot algorithm. *Math. Oper. Res.* 43(3):996–1024.

[26] Goldberg CH, West DB (1985) Bisection of circle colorings. *SIAM J. Algebraic Discrete Methods* 6(1):93–106.
[27] Göös M, Kamath P, Sotiraki K, Zampetakis M (2020) On the complexity of modulo-$q$ arguments and the Chevalley-Warning theorem. *Proc. 35th Comput. Complexity Conf.*, 1–42.
[28] Gourvès L (2019) Agreeable sets with matroidal constraints. *J. Combin. Optim.* 37(3):866–888.
[29] Hobby CR, Rice JR (1965) A moment problem in L1 approximation. *Proc. Amer. Math. Soc.* 16(4):665–670.
[30] Hollender A (2021) The classes PPA-$k$: Existence from arguments modulo $k$. *Theoretical Comput. Sci.* 885:15–29.
[31] Kyropoulou M, Suksompong W, Voudouris AA (2020) Almost envy-freeness in group resource allocation. *Theoretical Comput. Sci.* 841:110–123.
[32] Manurangsi P, Suksompong W (2017) Asymptotic existence of fair divisions for groups. *Math. Social Sci.* 89:100–108.
[33] Manurangsi P, Suksompong W (2019) Computing a small agreeable set of indivisible items. *Artificial Intelligence* 268:96–114.
[34] Manurangsi P, Suksompong W (2021) Almost envy-freeness for groups: Improved bounds via discrepancy theory. *Proc. 30th Internat. Joint Conf. Artificial Intelligence*, 335–341.
[35] Moulin H (2003) *Fair Division and Collective Welfare* (MIT Press).
[36] Moulin H (2007) Minimizing the worst slowdown: Offline, online. *Oper. Res.* 55(5):876–889.
[37] Neyman J (1946) Un théorème d'existence. *Comptes Rendus de l'Académie des sciences* 222:843–845.
[38] Oh H, Procaccia AD, Suksompong W (2021) Fairly allocating many goods with few queries. *SIAM J. Discrete Math.* 35(2):788–813.
[39] Papadimitriou CH (1994) On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. System Sci.* 48(3):498–532.
[40] Plaut B, Roughgarden T (2020) Almost envy-freeness with general valuations. *SIAM J. Discrete Math.* 34(2):1039–1068.
[41] Roughgarden T (2016) *Twenty Lectures on Algorithmic Game Theory* (Cambridge University Press).
[42] Rubinstein A (2018) Inapproximability of Nash equilibrium. *SIAM J. Comput.* 47(3):917–959.
[43] Sandomirskiy F, Segal-Halevi E (2019) Fair division with minimal sharing. Preprint, submitted August 5, https://arxiv.org/abs/1908.01669.
[44] Schuldenzucker S, Seuken S (2019) Monotonic and non-monotonic solution concepts for generalized circuits. Preprint, submitted July 30, https://arxiv.org/abs/1907.12854.
[45] Segal-Halevi E (2018) Fairly dividing a cake after some parts were burnt in the oven. *Proc. 17th Internat. Conf. Autonomous Agents Multiagent Systems*, 1276–1284.
[46] Segal-Halevi E, Nitzan S (2019) Fair cake-cutting among families. *Soc. Choice Welfare* 53(4):709–740.
[47] Segal-Halevi E, Suksompong W (2019) Democratic fair allocation of indivisible goods. *Artificial Intelligence* 277:103167.
[48] Segal-Halevi E, Suksompong W (2021) How to cut a cake fairly: A generalization to groups. *Amer. Math. Monthly* 128(1):79–83.
[49] Segal-Halevi E, Hassidim A, Aumann Y (2020) Envy-free division of land. *Math. Oper. Res.* 45(3):896–922.
[50] Simmons FW, Su FE (2003) Consensus-halving via theorems of Borsuk-Ulam and Tucker. *Math. Social Sci.* 45(1):15–25.
[51] Stromquist W, Woodall DR (1985) Sets on which several measures agree. *J. Math. Anal. Appl.* 108(1):241–248.
[52] Suksompong W (2016) Assigning a small agreeable set of indivisible items to multiple players. *Proc. 25th Internat. Joint Conf. Artificial Intelligence*, 489–495.
[53] Suksompong W (2018) Approximate maximin shares for groups of agents. *Math. Social Sci.* 92:40–47.
[54] Vondrák J (2010) CS 369P: Polyhedral techniques in combinatorial optimization, Lecture 17. Accessed February 5, 2022, http://theory.stanford.edu/~jvondrak/CS369P/lec17.pdf.