# Sublinear Approximation Algorithm for Nash Social Welfare with XOS Valuations

**Siddharth Barman**
Indian Institute of Science, Bangalore, India

**Anand Krishna**
Indian Institute of Science, Bangalore, India

**Pooja Kulkarni**
University of Illinois at Urbana-Champaign, IL, USA

**Shivika Narang**
Simons Laufer Mathematical Sciences Institute, Berkeley, CA, USA

## Abstract

We study the problem of allocating indivisible goods among $n$ agents with the objective of maximizing Nash social welfare (NSW). This welfare function is defined as the geometric mean of the agents' valuations and, hence, it strikes a balance between the extremes of social welfare (arithmetic mean) and egalitarian welfare (max-min value). Nash social welfare has been extensively studied in recent years for various valuation classes. In particular, a notable negative result is known when the agents' valuations are complement-free and are specified via value queries: for XOS valuations, one necessarily requires exponentially many value queries to find any sublinear (in $n$) approximation for NSW. Indeed, this lower bound implies that stronger query models are needed for finding better approximations. Towards this, we utilize demand oracles and XOS oracles; both of these query models are standard and have been used in prior work on social welfare maximization with XOS valuations.

We develop the first sublinear approximation algorithm for maximizing Nash social welfare under XOS valuations, specified via demand and XOS oracles. Hence, this work breaks the $O(n)$-approximation barrier for NSW maximization under XOS valuations. We obtain this result by developing a novel connection between NSW and social welfare under a capped version of the agents' valuations. In addition to this insight, which might be of independent interest, this work relies on an intricate combination of multiple technical ideas, including the use of repeated matchings and the discrete moving knife method. In addition, we partially complement the algorithmic result by showing that, under XOS valuations, an exponential number of demand and XOS queries are necessarily required to approximate NSW within a factor of $\left(1 - \frac{1}{e}\right)$.

**2012 ACM Subject Classification** Theory of computation → Approximation algorithms analysis

**Keywords and phrases** Discrete Fair Division, Nash Social Welfare, XOS Valuations

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2024.8

**Related Version** *Full Version*: `https://arxiv.org/abs/2110.00767`

**Funding** *Siddharth Barman*: The author is supported by a SERB Core research grant (CRG/2021/006165).
*Shivika Narang*: This work was conducted while the author was a PhD student at the Indian Institute of Science, Bengaluru, India and supported by Tata Consultancy Services research fellowship.

## 1 Introduction

The theory of fair division has been extensively studied over the past several decades in mathematical economics [9, 27] and, more recently, in computer science [10]. At the core of this vast body of work lies the question of finding fair and economically efficient allocations.

Through the years and for various settings, different notions of fairness and economic efficiency have been defined [27]. In particular, social welfare (defined as the sum of the valuations of the agents) is a standard measure of economic efficiency. On the other hand, egalitarian welfare (defined as the minimum value across the agents) is a well-established fairness criterion. Indeed, these two welfare objectives are not necessarily compatible; an allocation with high social welfare can have very low egalitarian welfare, and vice versa. A meaningful compromise between the extremes of economic efficiency and fairness is achieved through the Nash social welfare (NSW). This welfare function is defined as the geometric mean of the agents' valuations and it strikes a balance between between the arithmetic mean (average social welfare) and the minimum value (egalitarian welfare).

The Nash social welfare is known to satisfy fundamental axioms, including the Pigou-Dalton transfer principle, Pareto dominance, symmetry, and independence of unconcerned agents [27]. In fact, up to standard transformations, NSW is characteristically the unique welfare function that satisfies scale invariance along with particular fairness axioms [27]. The efficiency and fairness properties of NSW have been studied in context of both divisible and indivisible goods [24, 23, 28, 11]. Specifically, in the context of indivisible goods and additive valuations, [11] shows that any allocation that maximizes NSW is guaranteed to be envy-free up to one good.

**Significance of Approximating Nash Social Welfare.** As mentioned previously, NSW stands on axiomatic foundations. In particular, the Pigou-Dalton principle ensures that NSW will increase by transferring, say, $\delta$ value from a well-off agent $i$ to an agent $j$ with lower current value. At the same time, if the relative increase in $j$'s value is much less than the drop experienced by agent $i$, then NSW will not favor such a transfer, i.e., this welfare function also accommodates for collective efficiency. From a welfarist perspective, NSW induces a cardinal ordering (ranking) among the allocations and a meaningful goal is to find an allocation with as high a Nash social welfare as possible. This viewpoint is standard in cardinal treatments: each agent prefers bundles with higher values, and the social planner prefers allocations (valuation profiles) with higher welfare (be it social, Nash, or egalitarian). Indeed, this objective pervades all welfare functions–approximating NSW (in fair division contexts) is as well motivated as approximating social welfare (when economic efficiency is of central concern). Furthermore, it is important to note that, while Nash optimal allocations might satisfy additional fairness guarantees, this fact does not undermine the relevance of finding allocations with as high an NSW as possible. Overall, computing allocations with high NSW is a well-justified objective in and of itself. These observations, in particular, motivate the study of approximation algorithms for NSW maximization.

The current work addresses the problem of allocating *indivisible* goods with the aim of maximizing Nash social welfare. We focus on fair division instances wherein the agents' valuations are XOS functions. Specifically, a set function $v$ is said to be XOS (fractionally subadditive) iff it is a pointwise maximizer of additive functions, i.e., iff there exists a family of additive functions $\mathcal{F}$ such that $v(S) = \max_{f \in \mathcal{F}} f(S)$, for all subsets $S$. XOS functions constitute an encompassing class in the hierarchy of complement free valuations. This hierarchy has been extensively studied in the context of social welfare maximization [29] and includes, in order of containment, the following valuation classes: additive, submodular, XOS, and subadditive. As detailed below, these function families have also been the focus of recent works on Nash social welfare maximization.

**Computational results for NSW maximization.** In the indivisible-goods context and for additive valuations, a series of notable works have developed constant-factor approximation algorithms for the NSW maximization problem. The formative work of Cole and Gkatzelis [14] obtained the first constant-factor approximation (specifically, $2e^{1/e}$) for maximizing NSW under additive valuations. With an improved analysis, an approximation ratio of 2 for the problem was obtained in [13]. Also, an $e$-approximation has been achieved [1]; this result utilizes real stable polynomials. Currently, the best-known approximation ratio for additive valuations is $e^{1/e}$ [6].

Complementary to these positive results, the work of Garg et al. [19] shows that, under additive valuations, it is NP-hard to approximate NSW within a factor of 1.069; see also [25]. Furthermore, under submodular valuations, Garg et al. [22] showed that achieving an approximation ratio better than $e/(e-1)$ for NSW maximization – in the value-oracle model – is NP-hard. In the context of additive-like valuations, a $(2.404+\epsilon)$-approximation guarantee is known for budgeted additive valuations [20] and a 2-approximation has been obtained for separable piecewise linear concave (SPLC) valuations [2].

For the broader class of submodular valuations, an $O(n \log n)$ approximation guarantee was achieved in [22]; we will, throughout, use $n$ to denote the number of agents in the fair division instance. Furthermore, the recent work of Li and Vondrák [26] develops a constant-factor approximation algorithm for NSW maximization under submodular valuations. This result builds upon the work of Garg et al. [21] that addresses Rado valuations.

An $O(n)$-approximation ratio was obtained, independently, in [4] and [12] for the two most general valuation classes in above-mentioned hierarchy. That is, for NSW maximization a linear approximation guarantee can be achieved under XOS and subadditive valuations. Note that these algorithmic results hold under the standard value-oracle model, i.e., they only require values of different subsets (say, via an oracle). In fact, the work of Barman et al. [4] shows that, in the value-oracle model, this linear approximation guarantee is the best possible: under XOS (and, hence, subadditive) valuations, one necessarily requires exponentially many value queries to find any sublinear (in $n$) approximation for NSW. This (unconditional) lower bound necessitates the use of stronger query models for breaking the $O(n)$-approximation barrier. Towards this, the current work utilizes demand oracles and XOS oracles. Both of these query models have been used in prior work on social welfare maximization under XOS and subadditive valuations [17, 18, 16]. In particular, the work of Feige [18] uses demand oracles and achieves an $e/(e-1)$-approximation ratio for social welfare maximization under XOS valuations.

Note that, for an XOS valuation $v$ defined (implicitly) by a family of additive functions $\mathcal{F}$, an XOS oracle, when queried with a subset $S$, returns a maximizing additive function $f \in \mathcal{F}$, i.e., the oracle returns $f \in \arg\max_{f' \in \mathcal{F}} f'(S)$. Also, a demand oracle for valuation $v$ takes as input prices $p_g \in \mathbb{R}$, for all the goods $g$, and returns a set $S \subseteq [m]$ that maximizes $v(S) - \sum_{g \in S} p_g$.

## 1.1 Our Results and Techniques

We develop the first sublinear approximation algorithm for maximizing Nash social welfare under XOS valuations, specified via demand and XOS oracles.

**Main Result.** Given XOS and demand oracle access to the (XOS) valuations of $n$ agents, one can compute in polynomial-time (and with high probability) an $O(n^{53/54})$ approximation for the Nash social welfare maximization problem.

Our algorithm (Algorithm 1 in Section 3) first finds a linear (in $n$) approximation using essentially half the goods (obtained via random selection). This linear guarantee is achieved using intricate extensions of the idea of repeated matchings [22] and the discrete moving knife method [4]. A key contribution of the work is to then develop a novel connection between NSW and social welfare under a capped version of the agents' valuations. In particular, we use the linear guarantees as benchmarks and define capped valuations for the agents. Subsequently, we partition the remaining goods to (approximately) maximize social welfare under these capped valuations. We show that (for a relevant subset of agents) these steps bootstrap the linear guarantee into a sublinear bound. Indeed, this connection between social welfare, under the capped valuations, and NSW might be of independent interest. We also note that maximizing social welfare under capped valuations (with oracle access to the agents' underlying valuations and not the capped ones) is an involved step in and of itself. We use multiple other techniques to overcome such hurdles and overall obtain a sublinear approximation ratio through a sophisticated analysis. Section 3 provides a detailed overview of the algorithm and the main result is established in Section 5.

Furthermore, we complement, in part, the algorithmic result by showing that, under XOS valuations, an exponential number of demand and XOS queries are necessarily required to approximate NSW within a factor of $\left(1 - \frac{1}{e}\right)$; see Theorem 2 in Section 6 of the full version [5]. This unconditional lower bound is obtained by establishing a communication complexity result: considering (for analysis) a setting wherein each agent holds her XOS valuation, we show that exponential communication among the agents is required for approximating NSW within a factor of $\left(1 - \frac{1}{e}\right)$. Therefore, the query bound here holds not only for demand, XOS, and value queries, but applies to any (per-valuation) query model in which the queries and the oracle responses are polynomially large.

## 1.2 Additional Related Work

Nash social welfare maximization – specifically in the context of indivisible goods – has been prominently studied in recent years. Along with above-mentioned works, algorithmic results have also been developed for various special cases. In particular, the NSW maximization problem admits a polynomial-time (exact) algorithm for binary additive [15, 7] and binary submodular (i.e., matroid-rank) valuations [3]. Considering the particular case of binary XOS valuations, Barman and Verma [8] show that a constant-factor approximation for NSW maximization can be efficiently computed in the value-oracle model. They also prove that, by contrast, for binary subadditive valuations, any sublinear approximation requires an exponential number of value queries. Nguyen and Rothe [28] study instances with identical, additive valuations and develop a PTAS for maximizing NSW in such settings.

In contrast to the above-mentioned results, the current work addresses the entire class of XOS valuations and obtains a nontrivial approximation guarantee.

# 2 Notation and Preliminaries

We study the problem of discrete fair division, wherein $m$ indivisible goods have to be partitioned among $n$ agents in a fair manner. The cardinal preferences of the agents $i \in [n]$ (over a subset of goods) are specified via valuations $v_i \colon 2^{[m]} \mapsto \mathbb{R}_+$, where $v_i(S) \in \mathbb{R}_+$ is the value that agent $i$ has for subset of goods $S \subseteq [m]$. We denote an instance of a fair division problem by the tuple $\langle [n], [m], \{v_i\}_{i \in [n]} \rangle$.

This work focuses on XOS valuations. A set function $v \colon 2^{[m]} \mapsto \mathbb{R}_+$ is said to be XOS (or fractionally subadditive), iff there exists a family of additive set functions $\mathcal{F}$ such that, for each subset $S \subseteq [m]$, the value $v(S) = \max_{f \in \mathcal{F}} f(S)$. Note that the cardinality of the family $\mathcal{F}$ can be exponentially large in $m$. XOS valuations form a subclass of subadditive valuations; in particular, they satisfy $v(A \cup B) \leq v(A) + v(B)$, for all subsets $A, B \subseteq [m]$. We use $v_i(g)$ as a shorthand for $v_i(\{g\})$, i.e., for the value of good $g \in [m]$ for agent $i \in [n]$.

Since explicitly representing valuations (set functions) may require exponential space, prior works develop efficient algorithms assuming oracle access to the valuations. A basic oracle access is obtained through value queries: a value oracle, when queried with a subset $S$ returns the value of $S$. The current work uses demand oracles and XOS oracles. For an XOS valuation $v$ defined (implicitly) by a family of additive functions $\mathcal{F}$, an XOS oracle, when queried with a subset $S$, returns a maximizing additive function $f \in \mathcal{F}$, i.e., the oracle returns $f \in \arg\max_{f' \in \mathcal{F}} f'(S)$. Note that such an additive function $f$ can be completely specified by listing the values $\{f(g)\}_{g \in [m]}$. Also, given that $v$ is XOS and $f \in \mathcal{F}$, we have $v(T) \geq f(T)$, for all subsets $T \subseteq [m]$.

A demand oracle for valuation $v$ takes as input a price vector $p = (p_1, p_2, \ldots, p_m) \in \mathbb{R}^m$ over the $m$ goods (i.e., a demand query) and returns a set $S \subseteq [m]$ that maximizes $v(S) - \sum_{g \in S} p_g$. It is known that a demand oracle can simulate a value oracle (via a polynomial number of demand queries), but the converse is not true [29].

We will, throughout, assume that the agents' valuations $v_i$s are normalized ($v_i(\emptyset) = 0$) and monotone: $v_i(A) \leq v_i(B)$ for all $A \subseteq B \subseteq [m]$.

An allocation $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ is an $n$-partition of the $m$ indivisible goods, wherein subset $A_i$ is assigned to agent $i \in [n]$. Each such allocated subset $A_i \subseteq [m]$ will be referred to as a bundle. The goal of this work is to find allocations with as high a Nash social welfare as possible. Specifically, for a fair division instance $\langle [n], [m], \{v_i\}_{i \in [n]} \rangle$, the Nash social welfare, $\mathrm{NSW}(\cdot)$, of an allocation $\mathcal{A}$ is the geometric mean of the agents' valuations under $\mathcal{A}$, i.e., $\mathrm{NSW}(\mathcal{A}) \coloneqq \left(\prod_{i=1}^{n} v_i(A_i)\right)^{1/n}$. Throughout, we will write $\mathcal{N} = (N_1, \ldots, N_n)$ to denote an allocation that maximizes the Nash social welfare in the given instance and will refer to such an allocation as a Nash optimal allocation. In addition, let $g_i^*$ denote the good most valued by agent $i$ in the bundle $N_i$, i.e., $g_i^* \in \arg\max_{g \in N_i} v_i(g)$. We will assume, throughout, that $\mathrm{NSW}(\mathcal{N}) > 0$ and, hence, $v_i(g_i^*) > 0$. For the complementary case, wherein $\mathrm{NSW}(\mathcal{N}) = 0$, returning an arbitrary allocation suffices. With parameter $\alpha \geq 1$, an allocation $\mathcal{A} = (A_1, \ldots, A_n)$ is said to be an $\alpha$-approximate solution for the problem of maximizing Nash social welfare iff $\mathrm{NSW}(\mathcal{A}) \geq \frac{1}{\alpha} \mathrm{NSW}(\mathcal{N})$.

For subsets $S, T \subseteq [m]$, we will use the shorthand $S + T \coloneqq S \cup T$ and $S - T \coloneqq S \setminus T$. Furthermore, for good $g \in [m]$, we will write $S + g$ to denote $S + \{g\}$.

## 3 Algorithm and Technical Overview

Our main algorithm (Algorithm 1) consists of the following four phases:

**(I)** Keep aside a set of high-valued goods $\mathrm{M}$ (via the for-loop in Steps 2 to 5) and allocate $n$ goods via a matching, $\pi$ between set of agents $[n]$ and leftover goods, $[m] \setminus \mathrm{M}$ (in Step 6).

**(II)** Randomly partition the remaining goods into two parts, $\mathrm{R}$ and $\mathrm{R}'$.

**(III)** Allocate the subset of goods $\mathrm{R}$ among the agents – as bundles $X_1, X_2, \ldots, X_n$ – via a discrete moving knife procedure.

**(IV)** From the goods in $\mathrm{R}'$, find an allocation $(Y_1, \ldots, Y_n)$ that (approximately) maximizes social welfare under (judiciously defined) *capped valuations.*

**Phase I.** In the first phase, the algorithm identifies a subset of goods $\mathrm{M}$ that are kept aside while executing the intermediate phases. The algorithm rematches within $\mathrm{M}$ before termination (Step 12). The algorithm populates the set $\mathrm{M}$ by repeatedly finding matchings: it initializes $\mathrm{M} = \emptyset$, $\mathcal{G} = [m]$, and considers the complete weighted bipartite graph between the set of agents $[n]$ and the set of goods $\mathcal{G}$. The edge between agent $i$ and good $g$ has weight

**Algorithm 1** Sublinear approximation for Nash social welfare under XOS valuations.

**Input:** Instance $\langle [n], [m], \{v_i\}_{i \in [n]} \rangle$ with demand and XOS oracle access to the (XOS) valuations $v_i$s
**Output:** Allocation $\mathcal{Q} = (Q_1, \ldots, Q_n)$

1: Initialize $\mathrm{M} = \emptyset$ and $\mathcal{G} = [m]$
2: **for** $t = 1$ to $\log n$ **do**
3: Find matching $\tau_t : [n] \to \mathcal{G}$ that maximizes $\prod_{i \in [n]} v_i \left( \tau_t(i) \right)$
4: Update $\mathcal{G} \leftarrow \mathcal{G} - \{\tau_t(i)\}_{i \in [n]}$ and $\mathrm{M} \leftarrow \mathrm{M} + \{\tau_t(i)\}_{i \in [n]}$
5: **end for**
6: Find a matching $\pi : [n] \to \mathcal{G}$ that maximizes $\prod_{i \in [n]} v_i(\pi(i))$
7: Update $\mathcal{G} \leftarrow \mathcal{G} - \{\pi(i)\}_{i \in [n]}$
8: Randomly partition the set of goods $\mathcal{G}$ into R and R′, i.e., each good in $\mathcal{G}$ is included in R, or R′, independently with probability 1/2.
9: Set allocation $(X_1, X_2, \ldots, X_n) = \textsc{DiscreteMovingKnife}\left( [n], \mathrm{R}, \{v_i\}_{i \in [n]} \right)$ {This subroutine is detailed in Section 4.3}
10: For each $i \in [n]$, set (scaling factor) $\beta_i \coloneqq \frac{1}{n} \cdot \frac{1}{v_i(X_i + \pi(i))}$
11: Set allocation $(Y_1, Y_2, \ldots, Y_n) = \textsc{CappedSocialWelfare}\left( [n], \mathrm{R}', \{v_i\}_{i \in [n]}, (\beta_i)_{i \in [n]} \right)$ {This subroutine is detailed in Section 4.4}
12: Find matching $\mu : [n] \to \mathrm{M}$ that maximizes $\prod_{i \in [n]} v_i \left( \mu(i) + \pi(i) + X_i + Y_i \right)$ {Note that $\mu$ assigns to each agent a good from the set M, which was populated in the for-loop.}
13: **return** allocation $(Q_i = \mu(i) + \pi(i) + X_i + Y_i)_{i \in [n]}$

$w(i, g) = \log v_i(g)$. The algorithm then computes a maximum-weight matching $\tau$ in this bipartite graph and includes the matched goods $\{\tau(i)\}_{i \in [n]}$ in M (i.e., $\mathrm{M} \leftarrow \mathrm{M} + \{\tau(i)\}_{i \in [n]}$). Removing the matched goods from consideration, $\mathcal{G} \leftarrow \mathcal{G} - \{\tau(i)\}_{i \in [n]}$, the algorithm repeats this procedure $\log n$ times.

We will show that the set M contains, for each agent $i$, a distinct good of value at least $v_i(g_i^*)$ (see Lemma 3); recall that $g_i^*$ is the most-valued good in agent $i$'s optimal bundle, $g_i^* \in \arg\max_{g \in N_i} v_i(g)$. At a high level, this property will be used to establish approximation guarantees for agents $i$ that receive sufficiently high value via just the single good $g_i^*$.

The algorithm also seeds the allocation of each agent by finding a matching $\pi$ (in Step 6) – between $[n]$ and the goods $[m] \setminus \mathrm{M}$ – that maximizes the product (equivalently, the geometric mean) of the valuations. Each agent $i$ is permanently assigned the good $\pi(i)$.

Here, if the product of the values is zero ($\prod_{i=1}^{n} v_i(\pi(i)) = 0$), we consider matchings that maximize the number of agents who achieve a nonzero value (i.e., consider maximum-cardinality matchings with nonzero values) and among them select the one that maximizes the product.[1] Furthermore, all agents $z \in [n]$ with $v_z(\pi(z)) = 0$ will be excluded from consideration till Step 12 of the algorithm, i.e., such agents $z$ will not participate in phases three and four. For ease of presentation, we assume that there are no such agents (i.e., $v_i(\pi(i)) > 0$ for all $i$). This assumption does not affect the approximation guarantee; see the remark at the end of Section 5.1. Furthermore, the assumption ensures that parameters $\beta_i$s considered in the fourth phase (Step 10) are well-defined.

[1] Such a matching can be computed efficiently as a maximum-cardinality maximum-weight matching in the agents-goods bipartite graph; here the edge weight between agent $i$ and good $g$ is set to be $\log v_i(g)$, for nonzero $v_i(g)$s.

**Phase II.** The remaining goods $\mathcal{G} = [m] \setminus (\mathrm{M} + \pi([n]))$ are partitioned randomly into two subsets R and R$'$. Intuitively, the first phase addresses agents $i$ for whom good $g_i^* \in N_i$ by itself achieves a sublinear guarantee. Complementarily, the random partitioning and the subsequent phases are essentially aimed at agents $j$ for whom all the goods in $N_j$ are of sufficiently small value. This small-valued goods property (specifically, $v_j(g) \leq \frac{1}{\sqrt{n}} v_j(N_j)$, for all $g \in N_j$) ensures that, with high probability and for relevant agents $j$, both the values $v_j(N_j \cap \mathrm{R})$ and $v_j(N_j \cap \mathrm{R}')$ are within a constant factor of $v_j(N_j)$. We prove this via concentration bounds (see Lemma 4). Phases III and IV utilize the subsets of goods R and R$'$, respectively. The fact that, for relevant agents $j$, a near-optimal bundle exists in both R and R$'$ enables us to (a) obtain a linear approximation for the concerned agents in Phases III and (b) bootstrap the linear guarantee to a sublinear one in Phase IV. Notably, the current bootstrapping method goes beyond the substantial collection of techniques that have been recently developed for NSW maximization and it might be applicable in other resource-allocation contexts.

**Phase III.** This phase partitions the subset of goods R. As mentioned above, our aim here is to obtain a linear approximation for an appropriate set of agents. Towards that, we consider agents $i$ with the property that $v_i(g) \lesssim \frac{1}{n} v_i(\widetilde{N}_i)$, for a near-optimal bundle $\widetilde{N}_i$ and all goods $g \in \widetilde{N}_i$; see Section 4.3 for details. Now, to achieve a linear approximation guarantee for such agents $i$, we develop a discrete moving knife subroutine (Algorithm 2 in Section 4.3). Moving knife methods, in general, start with agent-specific (value) thresholds and then iteratively assign bundles (to the agents) that satisfy these thresholds. In the current context and to address the relevant subset of agents, we first restrict the valuation of each agent $j$ to the subset of goods that are individually of small value (for $j$) and then execute the moving knife method. This modification ensures that the developed subroutine finds an allocation $(X_1, \ldots, X_n)$ with the desired linear approximation guarantee.

**Phase IV.** A distinguishing idea in the current work is to use $v_i(X_i)$s (which provide a linear approximation for a relevant subset of agents) as a benchmark and bootstrap towards the desired sublinear bound in this phase. In particular, we use the values achieved in the allocation computed in Phase III – i.e., in $(X_1, \ldots, X_n)$ – to define, for each agent $i$, a scaling factor $\beta_i \coloneqq \frac{1}{n} \cdot \frac{1}{v_i(X_i + \pi(i))}$ (Step 10). Furthermore, we consider (capped) valuation $\widehat{v}_i(T) \coloneqq \min\left\{\frac{1}{\sqrt{n}},\ \beta_i v_i(T)\right\}$, for all subsets $T \subseteq \mathrm{R}'$.

The algorithm partitions the remaining subset of goods R$'$ in Step 11 by executing the subroutine CappedSocialWelfare. The objective of the subroutine is to (approximately) maximize social welfare under $\widehat{v}_i$s. Intuitively, the parameters $\beta_j$s are set to ensure that, when maximizing social welfare under $\widehat{v}_j$s, one prefers agents $i$ for whom $v_i(X_i + \pi(i))$ is much smaller than their optimal value $v_i(N_i)$. In Section 4.4, we detail the CappedSocialWelfare subroutine and show that its computed allocation $(Y_1, \ldots, Y_n)$ bootstraps the approximation guarantee towards the desired sublinear bound.

Note that maximizing social welfare is usually not aligned with the goal of maximizing NSW; an allocation with high social welfare can leave a small subset of agents with zero value and, hence, such an allocation would have zero Nash social welfare. Hence, approximating Nash social welfare by approximately maximizing social welfare (under capped valuations) is an interesting connection. This connection builds on the intricate guarantees obtained in the other phases and, in particular, it entails: $(i)$ carefully defining the capped valuations functions $\widehat{v}_i$s such that (a non-trivial fraction of) agents who have received low value in previous phases receive a high value in this phase, and $(ii)$ maximizing social welfare under

$\widehat{v}_i$s, with oracle access to the underlying valuations $v_i$s. Here, requirement $(ii)$ is nontrivial, since oracles for the valuations, $v_i$s, need to be appropriately modified to address $\widehat{v}_i$s. Such a modification is simple for value oracles, but not for demand and XOS oracles. We develop subroutine CappedSocialWelfare (detailed in Section 4.4) that overcomes these challenges and returns the desired allocation $(Y_1, \ldots, Y_n)$.

After these four phases, the algorithm finds a matching $\mu$ into the set of goods M, which was initially kept aside. The matching $\mu$ maximizes the product of the valuations with offset $X_i + Y_i + \pi(i)$ (Step 12).[2] Finally, each agent $i \in [n]$ is assigned the bundle $Q_i \coloneqq \pi(i) + X_i + Y_i + \mu(i)$.

In Section 4, we detail the above-mentioned phases and establish relevant guarantees for each. The phases work in close conjunction with each other; detailed guarantees from earlier phases support the successful executions of the latter ones. Overall, an intricate analysis is required to obtain the desired sublinear approximation guarantee. We accomplish this in Section 5 and establish the approximation ratio for the returned allocation $\mathcal{Q} = (Q_1, \ldots, Q_n)$. Specifically, we prove the following theorem in Section 5.

▶ **Theorem 1** (Main Result.). *Given instance $\langle [n], [m], \{v_i\}_{i \in [n]} \rangle$, with XOS and demand oracle access to (monotone and XOS) valuations $v_i$s, Algorithm* 1 *computes (with high probability) an $O(n^{53/54})$ approximation to the optimal Nash social welfare.*

We complement our algorithmic result, in part, with a query complexity result. We prove that exponential communication is required to approximate NSW within a factor of $(1-1/e)$. This lower bound on communication complexity directly provides a commensurate lower bound under the considered query models, i.e., under value, demand, and XOS queries. To prove the negative result we reduce the problem of MultiDisjointness to that of maximizing Nash social welfare. MultiDisjointness is a well-studied problem in the communication complexity literature. In this problem we have $n$ players and each player $i \in [n]$ holds a subset $B_i$ of a ground set of elements $[t]$. It is known that distinguishing between the cases of totally intersecting (i.e., there is an element that is included in $B_i$ for all $i \in [n]$) and totally disjoint (i.e., $B_i \cap B_j = \emptyset$ for all $i \neq j$) requires $\Omega\left({}^{t}/_{n}\right)$ communication. We reduce this problem to NSW maximization with each agent $i$ holding her XOS valuation $v_i$; in the reduction $t$ dictates the number of additive functions that define the XOS valuation of each agent. The key idea here is to show that there exists XOS valuations such that in the totally intersecting case the optimal NSW is sufficiently high and, complementarily, in the totally disjoint case it is sufficiently low. That is, between the two underlying cases, the optimal NSW bears a multiplicative gap of at least $(1 - 1/e)$. Therefore, the reduction shows that approximating NSW within a factor of $(1 - 1/e)$ necessarily requires $\Omega\left({}^{t}/_{n}\right)$ communication. With an exponentially large $t$, we obtain the desired query lower bound. Formally, we establish the following theorem (proof deferred to the full version [5].)

▶ **Theorem 2.** *For fair division instances with XOS valuations and a fixed constant $\varepsilon \in (0, 1]$, exponentially many demand and XOS queries are necessarily required for finding an allocation with* NSW *at least $\left(1 - \frac{1}{e} + \varepsilon\right)$ times the optimal.*

[2] Note that the matchings in Steps 3, 6, and 12 can be efficiently computed by finding a maximum-weight matching in a bipartite graph between the agents and the relevant goods; here, the weight of each edge is set as the log of the appropriate value.

# 4 Phases of Algorithm 1

## 4.1 Phase I: Isolating High-Valued Goods via Repeated Matching

Recall that $\mathrm{M}$ denotes the set of goods identified in Phase I of Algorithm 1 (see the for-loop at Step 2). Also, $\mathcal{N} = (N_1, \ldots, N_n)$ denotes a Nash optimal allocation and $g_i^* \in \arg\max_{g \in N_i} v_i(g)$, for all agents $i \in [n]$. As mentioned previously, Algorithm 1 keeps the goods in $\mathrm{M}$ aside while executing intermediate phases and at the end rematches within $\mathrm{M}$ (Step 12). The following lemma shows that $\mathrm{M}$ admits a matching $h$ wherein each agent receives a good with value at least that of $g_i^*$. At a high level, this lemma will be used to establish an approximation guarantee for agents $i$ that receive sufficiently high value via just the single good $g_i^*$. The proof of this lemma is deferred to the full version [5].

▶ **Lemma 3.** *There exists a matching $h \colon [n] \mapsto \mathrm{M}$ such that $v_i(h(i)) \geq v_i(g_i^*)$, for all agents $i \in [n]$.*

## 4.2 Phase II: Randomly Partitioning Goods

The following lemma addresses (near-optimal) bundles $\overline{N}_i \subseteq [m]$ with no high-valued goods. For such bundles, the lemma shows that randomly partitioning the goods – into two subsets $\mathrm{R}$ and $\mathrm{R}'$ – preserves, with high probability, sufficient value of $\overline{N}_i$ in both the parts, i.e., both $v_i(\overline{N}_i \cap \mathrm{R})$ and $v_i(\overline{N}_i \cap \mathrm{R}')$ are comparable to $v_i(\overline{N}_i)$. Hence, for bundles with no high-valued goods, the lemma implies that one can obtain sufficiently high welfare (Nash and social) in $\mathrm{R}$ as well as $\mathrm{R}'$. The proof of the lemma is deferred to the full version [5].

▶ **Lemma 4.** *Let $\mathcal{G}$ be a set of indivisible goods, $v_i$ be the XOS valuation of an agent $i \in [n]$, and $\overline{N}_i \subseteq \mathcal{G}$ be a subset with the property that $\max_{g \in \overline{N}_i} v_i(g) \leq \frac{1}{\sqrt{n}} v_i(\overline{N}_i)$. Then, for a random partition of $\mathcal{G}$ into sets $\mathrm{R}$ and $\mathrm{R}'$, we have*

$$\Pr\left\{ v_i(\overline{N}_i \cap \mathrm{R}) \leq \frac{1}{3} v_i(\overline{N}_i) \right\} \leq \exp\left( -\frac{\sqrt{n}}{18} \right) \qquad \textit{and}$$
$$\Pr\left\{ v_i(\overline{N}_i \cap \mathrm{R}') \leq \frac{1}{3} v_i(\overline{N}_i) \right\} \leq \exp\left( -\frac{\sqrt{n}}{18} \right).$$

*Here, random subset $\mathrm{R} \subseteq \mathcal{G}$ is obtained by selecting each good in $\mathcal{G}$ independently with probability $1/2$, and $\mathrm{R}' \coloneqq \mathcal{G} - \mathrm{R}$.*

Applying union bound, we extend Lemma 4 for allocations $(\overline{N}_1, \ldots, \overline{N}_n)$ to obtain:

▶ **Lemma 5.** *Given a set of indivisible goods $\mathcal{G}$ along with XOS valuations $v_i$ for agents $i \in [n]$, and a partition $(\overline{N}_1, \ldots, \overline{N}_n)$ of $\mathcal{G}$, let subset $T \coloneqq \{ i \in [n] : \max_{g \in \overline{N}_i} v_i(g) \leq \frac{1}{\sqrt{n}} v_i(\overline{N}_i) \}$. Then, for a random partition of $\mathcal{G}$ into sets $\mathrm{R}$ and $\mathrm{R}'$, we have*

$$\Pr\left\{ v_i(\overline{N}_i \cap \mathrm{R}) \geq \frac{1}{3} v_i(\overline{N}_i), \textit{ for all } i \in T \right\} \geq 1 - n \exp\left( -\frac{\sqrt{n}}{18} \right) \qquad \textit{and}$$
$$\Pr\left\{ v_i(\overline{N}_i \cap \mathrm{R}') \geq \frac{1}{3} v_i(\overline{N}_i), \textit{ for all } i \in T \right\} \geq 1 - n \exp\left( -\frac{\sqrt{n}}{18} \right).$$

*Here, random subset $\mathrm{R} \subseteq \mathcal{G}$ is obtained by selecting each good in $\mathcal{G}$ independently with probability $1/2$, and $\mathrm{R}' \coloneqq \mathcal{G} - \mathrm{R}$.*

## 4.3 Phase III: Discrete Moving Knife

This section presents the DiscreteMovingKnife subroutine (Algorithm 2). As mentioned previously, the subroutine is designed to address agents $i \in [n]$ for whom there exists (near-optimal) bundles $\widetilde{N}_i$ with the property that $\max_{g \in \widetilde{N}_i} \ v_i(g) \leq \frac{1}{16n} v_i(\widetilde{N}_i)$. Specifically, the subroutine obtains a linear approximation with respect to any allocation $(\widetilde{N}_1, \ldots, \widetilde{N}_n)$ and for the corresponding set of agents $T \coloneqq \{j \in [n] \ : \ \max_{g \in \widetilde{N}_j} \ v_j(g) \leq \frac{1}{16n} v_j(\widetilde{N}_j)\}$. Indeed, the DiscreteMovingKnife subroutine does not explicitly require as input an (near-optimal) allocation $(\widetilde{N}_1, \ldots, \widetilde{N}_n)$.

Given a set of indivisible goods R to partition, the subroutine (Algorithm 2) first finds, for each agent $j \in [n]$, a subset of goods $G_j \subseteq \mathrm{R}$ that solely consists of small-valued goods, i.e., $G_j$ satisfies $v_j(g) < \frac{1}{16n} v_j(G_j)$ for all $g \in G_j$. The set $G_j$ is computed by iteratively removing goods that violate the small-value requirement (see the while-loop in Step 3 of Algorithm 2). Observe that, by construction, for each $j$, when the while-loop (Step 5) terminates the set $G_j$ satisfies $v_j(g) < \frac{1}{16n} v_j(G_j)$ for all $g \in G_j$. Also, note that, as $G_j$ shrinks in the while-loop, the value $v_j(G_j)$ decreases. However we show that, for any allocation $(\widetilde{N}_1, \ldots, \widetilde{N}_n)$ and any agent $i$ from the set $T \coloneqq \{j \in [n] \ : \ \max_{g \in \widetilde{N}_j} \ v_j(g) \leq \frac{1}{16n} v_j(\widetilde{N}_j)\}$, the computed $G_i$ still satisfies $G_i \supseteq \widetilde{N}_i$.

That is, to obtain a linear approximation for agents in $T$, it suffices to find an allocation $(X_1, \ldots, X_n)$ with the property that $v_i(X_i) \geq \frac{1}{16n} v_i(G_i)$ for all $i \in T$. The subsequent steps of the DiscreteMovingKnife subroutine find such an allocation. In particular, for each agent $j \in [n]$, the subroutine restricts attention to the set $G_j$, i.e., considers valuation $v'_j(S) \coloneqq v_j(S \cap G_j)$, for all subsets $S \subseteq \mathrm{R}$ (Step 7). This construction ensures that for each agent $j \in [n]$ and all goods $g \in \mathrm{R}$ we have $v'_j(g) \leq \frac{1}{16n} v'_j(G_j) = v'_j(\mathrm{R})$.

The subroutine then goes over all the goods in R in an arbitrary order and adds them one by one into a bundle $P$, until an agent $a$ calls out that her value (under $v'_a$) for $P$ is at least $\frac{1}{16n} v'_a(\mathrm{R})$. We assign these goods to agent $a$ and remove them (along with agent $a$) from consideration (Step 12). The subroutine iterates over the remaining set of agents and goods. Note that the subroutine only requires value-oracle access to the valuations $v_j$s.[3] The following lemma (proof in full version [5]) shows that the computed allocation $(X_1, X_2, \ldots, X_n)$ achieves the desired linear approximation.

▶ **Lemma 6.** *Let $\langle [n], \mathrm{R}, \{v_i\}_{i \in [n]} \rangle$ be a fair division instance with XOS valuations $v_i$s. Also, let $(\widetilde{N}_1, \ldots, \widetilde{N}_n)$ be any allocation with $T \coloneqq \left\{ i \in [n] \ : \ \max_{g \in \widetilde{N}_i} \ v_i(g) < \frac{1}{16n} v_i(\widetilde{N}_i) \right\}$. Then, given value-oracle access to $v_i$s, the DiscreteMovingKnife subroutine computes – in polynomial time – an allocation $(X_1, \ldots, X_n)$ with the property that $v_i(X_i) \geq \frac{1}{16n} v_i(\widetilde{N}_i)$ for all $i \in T$.*

## 4.4 Phase IV : Maximizing Capped Social Welfare

The section presents the CappedSocialWelfare subroutine (Algorithm 3) that maximizes social welfare under capped versions of the given valuations $v_i$s. Specifically, given a fair division instance $\langle [n], \mathrm{R}', \{v_i\}_{i \in [n]} \rangle$ and parameters $\beta_1, \ldots, \beta_n \in \mathbb{R}_+$, we define capped valuations, for each agent $i \in [n]$, as follows: $\widehat{v}_i(S) \coloneqq \min \left\{ \frac{1}{\sqrt{n}}, \ \beta_i v_i(S) \right\}$ for all subsets $S$[4].

---

[3] We can efficiently simulate the value oracle for $v'_j$ as follows: for any queried subset $S$, the value $v'_j(S)$ can be obtained by querying for $v_j(G_j \cap S)$.

[4] Algorithm 1 invokes the subroutine CappedSocialWelfare with $\beta_i = \frac{1}{n} \frac{1}{v_i(X_i + \pi(i))}$. However, the results obtained in this section hold for any positive $\beta_i$s.

**Algorithm 2** DiscreteMovingKnife.

**Input:** Instance $\langle [n], \mathrm{R}, \{v_i\}_{i \in [n]} \rangle$ with value-oracle access to the valuations $v_i$s
**Output:** An allocation $(X_1, X_2, \ldots, X_n)$

```
For each agent j ∈ [n], initialize set G_j = R
for j = 1 to n do
    while there exists a good g ∈ G_j such that v_j(g) ≥ (1/16n) v_j(G_j) do
        Update G_j ← G_j − {g}
    end while
end for
Define v'_j(S) := v_j(S ∩ G_j) for all agents j ∈ [n] and subsets S ⊆ R
Initialize set of goods Γ = R along with agents A = [n] and bundles X_j = ∅, for all
    j ∈ [n]. Also, set P = ∅.
while Γ ≠ ∅ and A ≠ ∅ do
    Pick an arbitrary good g ∈ Γ, and update P ← P + {g} along with Γ ← Γ − {g}
    if there exists an agent a ∈ A such that v'_a(P) ≥ (1/16n) v'_a(R) then
        Assign X_a = P and update A ← A − {a} along with P = ∅
    end if
end while
If Γ ≠ ∅, update X_n ← X_n + Γ
return allocation (X_1, X_2, ..., X_n).
```

Since the valuations $v_i$s are XOS, the functions $\widehat{v}_i$s are subadditive. Also, note that, using the value oracle for $v_i$, we can easily implement the value oracle for $\widehat{v}_i$. However, a key hurdle for the subroutine is that it does not have demand oracles for $\widehat{v}_i$s; otherwise, one could directly invoke the approximation algorithm of Feige [18] to maximize social welfare. We design the subroutine to overcome this hurdle and (approximately) maximize the social welfare under $\widehat{v}_i$s, using (XOS and demand) oracle access to $v_i$s.

Our approximation guarantee (for social welfare under capped valuations) holds for instances $\langle [n], \mathrm{R}', \{v_i\}_i \rangle$ wherein there exists an allocation $(O_1, \ldots, O_n)$ and a subset of agents $\overline{A} \subseteq [n]$ that satisfy

**P1:** The welfare $\sum_{i \in \overline{A}} \widehat{v}_i(O_i) \geq \frac{26}{27}\sqrt{n}$.

**P2:** For each agent $i \in \overline{A}$ and all goods $g' \in O_i$, the value $\widehat{v}_i(g') \leq \frac{1}{2\sqrt{n}}$.

When, in the analysis of the main algorithm (i.e., in Section 5), we invoke the guarantee obtained here we will show that these two properties hold for the instance at hand. Also, note that both the properties express conditions in terms of the capped valuations $\widehat{v}_i$s. In particular, Property **P2** states that, for each agent $i$ in the designated set $\overline{A}$, all the goods in the bundle $O_i$ are of sufficiently small value. Property **P1** demands that we have high enough welfare (under $\widehat{v}_i$) among the bundles $O_i$ assigned to agents $i \in \overline{A}$.

For XOS valuation $v_i$, let $\mathcal{F}_i$ denote the family of additive functions that define $v_i$. Throughout this section, we will write $f_{i,S}$ to denote the additive function in $\mathcal{F}_i$ that induces $v_i(S)$, i.e., for any subset $S$,

$$f_{i,S} := \arg\max_{f \in \mathcal{F}_i} f(S) \tag{1}$$

The subroutine CappedSocialWelfare (Algorithm 3) starts with empty bundles, $Y_i = \emptyset$ for all agents $i \in [n]$, and with the set of unallocated goods $Y_0 = \mathrm{R}'$. Throughout, $Y_0$ denotes the set of unallocated goods with the maintained allocations $(Y_1, \ldots, Y_n)$. The subroutine iteratively transfers goods from $Y_0$ and between bundles as long as an increase in

**Algorithm 3** CappedSocialWelfare.

**Input:** Instance $\mathcal{I} = \langle [n], \mathrm{R}', \{v_i\}_{i\in[n]} \rangle$, with demand and XOS oracle access to the valuations $v_i$s, and parameters $\{\beta_i\}_{i\in[n]}$

**Output:** Allocation $(Y_1, \ldots, Y_n)$

1: Initialize $Y_i = \emptyset$, for all $i \in [n]$, and $Y_0 = \mathrm{R}'$ {$Y_0$ is the set of unallocated goods}
2: Flag $\leftarrow$ **true**
3: **while** Flag **do**
4: For every good $g \in Y_0$ set price $p_g = 0$
5: For every agent $j \in [n]$ and bundle $Y_j$, query the XOS oracle for $v_j$ to find additive function $f_{j,Y_j}(\cdot)$. For each $g \in Y_j$, set price $p_g = 2\beta_j \ f_{j,Y_j}(g)$
6: **for** each agent $j \in [n]$ **do**
7: For each good $g \in \mathrm{R}'$ with $\widehat{v}_j(g) \leq \frac{1}{2\sqrt{n}}$, set price $q^j_g = p_g$
8: For each good $g \in \mathrm{R}'$ with $\widehat{v}_j(g) > \frac{1}{2\sqrt{n}}$, set price $q^j_g = \infty$
9: Let $D_j = \{g_1, \ldots, g_{|D_j|}\}$ be the demand set under valuation $v_j$ and prices $q^j_g/\beta_j$ {Set $D_j$ is obtained via the given demand oracle for $v_j$. The goods in this set, $g_1, \ldots, g_{|D_j|}$, are indexed in an arbitrary order.}
10: Let $k$ be the minimum index such that $\widehat{v}_j(\{g_1, \ldots g_k\}) \geq \frac{92}{225}\frac{1}{\sqrt{n}}$
11: Set $\widehat{D}_j = \{g_1, \ldots, g_k\}$ {In case $\widehat{v}_j(D_j) < \frac{92}{225\sqrt{n}}$, set $\widehat{D}_j = D_j$}
12: **end for**
13: **if** there exists an agent $a \in [n]$ such that $\widehat{v}_a(\widehat{D}_a) + \sum_{j\in[n]\setminus\{a\}} \widehat{v}_j(Y_j - \widehat{D}_a) \geq \sum_{j=1}^{n} \widehat{v}_j(Y_j) + \frac{1}{225\sqrt{n}}$ **then**
14: Assign $Y_a = \widehat{D}_a$
15: For all $j \in [n] \setminus \{a\}$, update $Y_j \leftarrow Y_j - \widehat{D}_a$.
16: Also, update the set of unallocated goods $Y_0 = \mathrm{R}' \setminus \left(\cup_{j=1}^{n} Y_j\right)$
17: **else**
18: Flag $\leftarrow$ **false**
19: **end if**
20: **end while**
21: **return** Allocation $(Y_1, \ldots, Y_n)$.

social welfare (with respect to $\widehat{v}_i$s) is obtained. Throughout its execution, the subroutine considers the efficacy of transferring a subset of goods $\widehat{D}_a$, to agent $a$, based on the current (social welfare) contribution of each good $g \in \widehat{D}_a$. In particular, for each agent $j$, we consider the contribution of the goods $g \in Y_j$ with respect to the additive function that induces $v_j(Y_j)$. Therefore, the sum of these contributions over $g \in Y_j$ is equal to $v_j(Y_j)$. For every good $g$ we set the price $p_g$ to be $2\beta_j$ times $g$'s contribution (see Step 5). The price of the unallocated goods is set to be zero. Then, bearing in mind property **P2**, we set agent-specific prices $q^j_g$s to ensure that for agent $j$ only goods with small-enough value are eligible for transfer (Steps 7 and 8). Scaling $q^j_g$s appropriately for each agent $j$, the algorithm finds a demand set $D_j$ under $v_j$ (Step 9). For each agent $j$, the candidate set $\widehat{D}_j$ is obtained by selecting a cardinality-wise minimal subset of $D_j$ of sufficiently high value (Steps 10 and 11). We will prove that, until the social welfare (under $\widehat{v}_j$s) of the maintained allocation $(Y_1, \ldots, Y_n)$ reaches a high-enough value, assigning $\widehat{D}_a$ to an agent $a$ (and removing the goods in $\widehat{D}_a$ from the other agents' bundles) increases the welfare (Step 13). That is, for any considered allocation $(Y_1, \ldots, Y_n)$ with social welfare less than a desired threshold, the if-condition in

Step 13 necessarily holds; see Lemma 10. This lemma will establish the main result of this section (Theorem 7 below) that lower bounds the social welfare – under $\widehat{v}_i$s – of the computed allocation.

▶ **Theorem 7.** *Let $\langle [n], \mathrm{R}', \{v_i\}_{i\in[n]}\rangle$ be a fair division instance in which there exists an allocation $(O_1, \ldots, O_n)$ and a subset of agents $\overline{A} \subseteq [n]$ that satisfy properties* **P1** *and* **P2** *mentioned above. Then, given XOS and demand oracle access to the (XOS) valuations $v_i$s, Algorithm 3 computes (in polynomial time) an allocation $(Y_1, \ldots, Y_n)$ such that*

$$\sum_{j=1}^{n} \widehat{v}_j(Y_j) \geq \frac{2}{25} \sum_{j\in\overline{A}} \widehat{v}_j(O_j).$$

Note that the welfare bound obtained in this theorem is with respect to the agents in $\overline{A}$. For our analysis, it suffices to have a guarantee of this form. It is, however, interesting to note that (under property **P1**) we also obtain a 13-approximation for the optimal social welfare: $\sum_{j=1}^{n} \widehat{v}_j(Y_j) \geq \frac{2}{25} \sum_{j\in\overline{A}} \widehat{v}_j(O_j) \geq \frac{2}{25}\frac{26\sqrt{n}}{27} \geq \frac{\sqrt{n}}{13}$; recall that, by definition, each function $\widehat{v}_j$ is upper bounded by $\frac{1}{\sqrt{n}}$ and, hence, the optimal social welfare under these functions is at most $\sqrt{n}$.

To prove Theorem 7, we first establish the following lemmas.

▶ **Lemma 8.** *Throughout its execution, Algorithm 3 maintains*

$$\widehat{v}_j(Y_j) = \beta_j v_j(Y_j) < \frac{1}{\sqrt{n}} \qquad \text{for all agents } j \in [n].$$

**Proof.** Fix any agent $j \in [n]$ and consider any iteration in which $j$ receives set $\widehat{D}_j$ (i.e., Step 14 executes with $a = j$). We will first show that $\widehat{v}_j(\widehat{D}_j) < \frac{1}{\sqrt{n}}$. Note that $\widehat{D}_j \subseteq D_j$, where $D_j$ is the demand set queried for agent $j$ in Step 9; in particular, $D_j \in \arg\max_S \left( v_j(S) - \sum_{g\in S} \frac{q_g^j}{\beta_j} \right)$. Hence, the goods in $D_j$ have finite prices, $q_g^j < \infty$. Consequently, for each $g \in D_j$, we have $\widehat{v}_j(g) \leq \frac{1}{2\sqrt{n}}$; see Steps 7 and 8. Furthermore, given that $\widehat{D}_j$ is a minimal set (within $D_j$) with value at least $\frac{92}{225\sqrt{n}}$ (Steps 10 and 11) and $\widehat{v}_j$ is subadditive, we obtain $\widehat{v}_j(\widehat{D}_j) \leq \frac{92}{225\sqrt{n}} + \frac{1}{2\sqrt{n}} < \frac{1}{\sqrt{n}}$.

This bound implies that throughout the subroutine's execution agent $j$ receives a bundle $Y_j$ of value (under $\widehat{v}_j$) less than $\frac{1}{\sqrt{n}}$: At the beginning of the subroutine $Y_j = \emptyset$, i.e., $\widehat{v}_j(Y_j) = 0$. Furthermore, between executions of Step 14 specifically for agent $j$, goods are only removed from $Y_j$. Therefore, monotonicity of the function $\widehat{v}_j$ ensures that $\widehat{v}_j(Y_j) < 1/\sqrt{n}$ throughout the execution Algorithm 3.

By definition, $\widehat{v}_j(Y_j) = \min\left\{\frac{1}{\sqrt{n}}, \beta_j v_j(Y_j)\right\}$. Hence, the inequality $\widehat{v}_j(Y_j) < \frac{1}{\sqrt{n}}$ gives us $\widehat{v}_j(Y_j) = \beta_j \ v_j(Y_j) < \frac{1}{\sqrt{n}}$. The lemma stands proved. ◀

The next lemma bounds the loss in welfare due to reassignment of goods in Step 15 of the subroutine.

▶ **Lemma 9.** *In any while-loop iteration of Algorithm 3, let $Y_j$ be the bundle assigned to any agent $j \in [n]$ and $p_g$s be the prices at the start of the iteration (i.e., in Step 5). Then, for all subsets $X \subseteq Y_j$*

$$\widehat{v}_j\left(Y_j \setminus X\right) \geq \widehat{v}_j(Y_j) - \frac{1}{2} \sum_{g\in X} p_g.$$

**Proof.** Note that $f_{j,Y_j}$ denotes the additive function that induces $v_j(Y_j)$ (see equation (1)). Therefore,

$$\widehat{v}_j(Y_j) = \beta_j v_j(Y_j) = \beta_j \sum_{g \in Y_j} f_{j,Y_j}(g) \qquad \text{(via Lemma 8 and definition of } f_{j,Y_j}\text{)}$$
$$= \frac{1}{2} \sum_{g \in Y_j} p_g \tag{2}$$

The last inequality follows from how the prices, $p_g$s, were set in Step 5 of Algorithm 3. Furthermore, using the fact that $v_j$ is XOS we obtain

$$\beta_j \ v_j(Y_j \setminus X) \geq \beta_j \sum_{g \in Y_j \setminus X} f_{j,Y_j}(g) = \frac{1}{2} \sum_{g \in Y_j \setminus X} p_g = \frac{1}{2} \sum_{g \in Y_j} p_g - \frac{1}{2} \sum_{g \in X} p_g \qquad \text{(by definition of } p_g \text{ in Step 5)}$$
$$= \widehat{v}_j(Y_j) - \frac{1}{2} \sum_{g \in X} p_g \qquad \text{(via (2))}$$

Since the function $\widehat{v}_j$ is monotone, for any subset $X \subseteq Y_j$, we have $\widehat{v}_j(Y_j \setminus X) \leq \widehat{v}_j(Y_j) < \frac{1}{\sqrt{n}}$; here, the last inequality follows from Lemma 8. Therefore, $\widehat{v}_j(Y_j \setminus X) = \beta_j \ v_j(Y_j \setminus X)$. These observations establish the desired inequality: $\widehat{v}_j(Y_j \setminus X) \geq \widehat{v}_j(Y_j) - \frac{1}{2} \sum_{g \in X} p_g$. ◀

The next lemma shows that, in Algorithm 3, the social welfare (under $\widehat{v}_j$s) of the maintained allocation $(Y_1, \ldots, Y_n)$ keeps on increasing till it reaches $\frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j)$. That is, for any considered allocation $(Y_1, \ldots, Y_n)$ with social welfare less than $\frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j)$, the if-condition in Step 13 necessarily holds.

▶ **Lemma 10.** *Let $(Y_1, \ldots, Y_n)$ be an allocation considered in any iteration of Algorithm 3 with the property that*

$$\sum_{j=1}^{n} \widehat{v}_j(Y_j) < \frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j).$$

*Then, there exists an agent $a \in [n]$ such that*

$$\widehat{v}_a(\widehat{D}_a) + \sum_{j \in [n] \setminus \{a\}} \widehat{v}_j(Y_j \setminus \widehat{D}_a) \geq \sum_{j=1}^{n} \widehat{v}_j(Y_j) + \frac{1}{225\sqrt{n}}.$$

*Here, set $\widehat{D}_a$ is as defined in Step 11 of the algorithm.*

**Proof.** First, we express the social welfare of allocation $(Y_1, \ldots, Y_n)$ in terms of the prices $p_g$s (set in Step 5)

$$\sum_{j=1}^{n} \widehat{v}_j(Y_j) = \sum_{j=1}^{n} \beta_j \ v_j(Y_j) = \sum_{j=1}^{n} \beta_j \sum_{g \in Y_j} f_{j,Y_j}(g) \qquad \text{(via Lemma 8 and definition of } f_{j,Y_j}\text{)}$$
$$= \sum_{j=1}^{n} \sum_{g \in Y_j} \frac{p_g}{2} = \sum_{g \in \mathrm{R}' - Y_0} \frac{p_g}{2} \qquad \text{(considering Step 5)}$$
$$= \sum_{g \in \mathrm{R}'} \frac{p_g}{2} \qquad (p_g = 0, \text{ for all } g \in Y_0; \text{ Step 4})$$

Therefore, the lemma assumption, $\sum_{j=1}^{n} \widehat{v}_j(Y_j) < \frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j)$, reduces to $\frac{1}{2} \sum_{g \in \mathrm{R}'} p_g < \frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j)$.

Multiplying both sides of this inequality by 2 gives us

$$\sum_{g \in \mathrm{R}'} p_g < \frac{4}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j) = \sum_{j \in \overline{A}} \widehat{v}_j(O_j) - \frac{21}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j) \leq \sum_{j \in \overline{A}} \widehat{v}_j(O_j) - \frac{21}{25} \left( \frac{26\sqrt{n}}{27} \right)$$

(from property **P1**)

$$\leq \sum_{j \in \overline{A}} \widehat{v}_j(O_j) - \frac{182\sqrt{n}}{225} \tag{3}$$

For any set $S$, write cumulative price $p(S) \coloneqq \sum_{g \in S} p_g$. Applying this notation and rearranging inequality (3) we get[5]

$$\sum_{j \in \overline{A}} \left( \widehat{v}_j(O_j) - p(O_j) \right) \geq \frac{182\sqrt{n}}{225} \tag{4}$$

Next, we define subset of agents $H \coloneqq \left\{ h \in [n] : \widehat{v}_h(Y_h) \geq \frac{1}{5\sqrt{n}} \right\}$.

The lemma assumption $\sum_{j=1}^{n} \widehat{v}_j(Y_j) < \frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j)$ implies that $|H| < \frac{2n}{5}$. Otherwise, we would obtain a contradiction: $\sum_{h \in H} \widehat{v}_h(Y_h) \geq \frac{|H|}{5\sqrt{n}} \geq \frac{2\sqrt{n}}{25} \geq \frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j)$. Recall that, by definition, the valuations $\widehat{v}_j$s are upper bounded by $\frac{1}{\sqrt{n}}$.

Inequality (4) can be expressed as

$$\sum_{j \in \overline{A} \setminus H} \left( \widehat{v}_j(O_j) - p(O_j) \right) + \sum_{h \in \overline{A} \cap H} \left( \widehat{v}_h(O_h) - p(O_h) \right) \geq \frac{182\sqrt{n}}{225}.$$

Therefore, the inequality $\widehat{v}_h(O_h) \leq \frac{1}{\sqrt{n}}$ and the fact that prices are nonnegative, lead to $\sum_{j \in \overline{A} \setminus H} \left( \widehat{v}_j(O_j) - p(O_j) \right) \geq \frac{182\sqrt{n}}{225} - \frac{|\overline{A} \cap H|}{\sqrt{n}} \geq \frac{182\sqrt{n}}{225} - \frac{|H|}{\sqrt{n}} \geq \frac{182\sqrt{n}}{225} - \frac{2\sqrt{n}}{5} = \frac{92\sqrt{n}}{225}$.

Hence, there exists an agent $a \in \overline{A} \setminus H$ such that

$$\widehat{v}_a(O_a) - p(O_a) \geq \frac{1}{|\overline{A} \setminus H|} \frac{92\sqrt{n}}{225} \geq \frac{92}{225\sqrt{n}} \tag{5}$$

We will complete the proof by showing that the lemma holds for this specific agent $a \in \overline{A} \setminus H$. Towards this, we first bound $\widehat{v}_a(\widehat{D}_a)$ and show that this value is at least the price of the set $\widehat{D}_a$.

Recall that set $D_a$ is obtained by the demand oracle for agent $a$ (i.e., for valuation $v_a$) under prices $\frac{q^a_g}{\beta_a}$ (Step 9). Furthermore, for the agent $a \in \overline{A} \setminus H$ and all goods $g' \in O_a$, we have $\widehat{v}_a(g') \leq \frac{1}{2\sqrt{n}}$ (via property **P2**). Hence, all goods $g' \in O_a$ have finite prices that satisfy $q^a_{g'} = p_{g'}$ (Step 7). Hence, $O_a$ a feasible set to be demanded and the demand optimality of $D_a$ gives us $v_a(D_a) - \sum_{g \in D_a} \left( \frac{p_g}{\beta_a} \right) \geq v_a(O_a) - \sum_{g \in O_a} \left( \frac{p_g}{\beta_a} \right)$. Multiplying throughout by $\beta_a > 0$ we obtain

$$\beta_a v_a(D_a) - \sum_{g \in D_a} p_g \geq \beta_a v_a(O_a) - \sum_{g \in O_a} p_g \geq \widehat{v}_a(O_a) - \sum_{g \in O_a} p_g \qquad \text{(By definition of } \widehat{v}_a\text{)}$$

$$\geq \frac{92}{225\sqrt{n}} \qquad \text{(via (5))}$$

[5] Recall that the prices $p_g$s are nonnegative.

Since the the prices are non-negative, $\beta_a v_a(D_a) \geq \frac{92}{225\sqrt{n}}$. That is, in Step 11 for agent $a$ the desired set $\widehat{D}_a \subseteq D_a$ can be found with value $\widehat{v}_a(\widehat{D}_a) \geq \frac{92}{225\sqrt{n}}$. In addition, the demand optimality of $D_a$ implies that all the goods $g \in D_a$ have finite prices. Therefore, $\widehat{v}_a(g) \leq \frac{1}{2\sqrt{n}}$ for all goods $g \in D_a \supseteq \widehat{D}_a$. This bound and the selection of $D_a$ gives us

$$\widehat{v}_a(\widehat{D}_a) \leq \frac{92}{225\sqrt{n}} + \frac{1}{2\sqrt{n}} < \frac{1}{\sqrt{n}} \tag{6}$$

We will next show that $\widehat{v}_a(\widehat{D}_a)$ is at least the price of the set $\widehat{D}_a$. Write $f_{a,D_a}(\cdot)$ to denote the additive function that induces $v_a(D_a)$; in particular, $v_a(D_a) = \sum_{g \in D_a} f_{a,D_a}(g)$. The demand optimality of $D_a$, under the prices $\frac{q_g^a}{\beta_a}$, implies $f_{a,D_a}(g) - \frac{q_g^a}{\beta_a} \geq 0$ for all goods $g \in D_a$. Equivalently, $\beta_a f_{a,D_a}(g) - q_g^a \geq 0$ for all goods $g \in D_a \supseteq \widehat{D}_a$. Using the bound we obtain

$$\begin{aligned}
\widehat{v}_a(\widehat{D}_a) - \sum_{g \in \widehat{D}_a} p_g &= \beta_a v_a(\widehat{D}_a) - \sum_{g \in \widehat{D}_a} p_g && \text{(via (6) and the definition of } \widehat{v}_a) \\
&\geq \sum_{g \in \widehat{D}_a} \beta_a f_{a,D_a}(g) - \sum_{g \in \widehat{D}_a} p_g && \text{(since } v_a \text{ is XOS)} \\
&= \sum_{g \in \widehat{D}_a} \left(\beta_a f_{a,D_a}(g) - p_g\right) \geq 0.
\end{aligned}$$

That is,

$$\widehat{v}_a(\widehat{D}_a) \geq \sum_{g \in \widehat{D}_a} p_g \tag{7}$$

Using 7 we can bound the change in social welfare when $\widehat{D}_a$ is assigned to agent $a$:

$$\begin{aligned}
\widehat{v}_a(\widehat{D}_a) - \widehat{v}_a(Y_a) + &\sum_{j \in [n] \setminus \{a\}} \Big(\widehat{v}_j(Y_j \setminus \widehat{D}_a) - \widehat{v}_j(Y_j)\Big) \\
&\geq \Big(\widehat{v}_a(\widehat{D}_a) - \widehat{v}_a(Y_a)\Big) - \frac{1}{2} \sum_{g \in \widehat{D}_a} p_g \geq \Big(\widehat{v}_a(\widehat{D}_a) - \widehat{v}_a(Y_a)\Big) - \frac{\widehat{v}_a(\widehat{D}_a)}{2} \\
& && \text{(via Lemma 9 and (7))} \\
&= \frac{1}{2}\widehat{v}_a(\widehat{D}_a) - \widehat{v}_a(Y_a) \geq \frac{92}{450\sqrt{n}} - \widehat{v}_a(Y_a) && \text{(via (6))} \\
&\geq \frac{92}{450\sqrt{n}} - \frac{1}{5\sqrt{n}} = \frac{1}{225\sqrt{n}}. && \text{(since } a \in \overline{A} \setminus H \text{, i.e., } a \notin H)
\end{aligned}$$

Hence, for agent $a$ we necessarily obtain the desired increase in social welfare:

$$\widehat{v}_a(\widehat{D}_a) + \sum_{j \in [n] \setminus \{a\}} \widehat{v}_j(Y_j \setminus \widehat{D}_a) \geq \sum_{j=1}^{n} \widehat{v}_j(Y_j) + \frac{1}{225\sqrt{n}}.$$

This completes the proof. ◀

We now restate and prove Theorem 7.

▶ **Theorem 7.** *Let $\langle [n], \mathrm{R}', \{v_i\}_{i \in [n]} \rangle$ be a fair division instance in which there exists an allocation $(O_1, \ldots, O_n)$ and a subset of agents $\overline{A} \subseteq [n]$ that satisfy properties* **P1** *and* **P2** *mentioned above. Then, given XOS and demand oracle access to the (XOS) valuations $v_i$s, Algorithm 3 computes (in polynomial time) an allocation $(Y_1, \ldots, Y_n)$ such that*

$$\sum_{j=1}^{n} \widehat{v}_j(Y_j) \geq \frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j).$$

**Proof.** The contrapositive of Lemma 10, implies that if there does not exist an agent $a$ such that $\widehat{v}_a(\widehat{D}_a) + \sum_{j \neq a} \widehat{v}_j(Y_j - \widehat{D}_a) \geq \sum_{j=1}^{n} \widehat{v}_j(Y_j) + \frac{1}{225\sqrt{n}}$ (i.e., the if-condition in Step 13 is *not* satisfied), then $\sum_{j=1}^{n} \widehat{v}_j(Y_j) \geq \frac{2}{25} \sum_{j \in \overline{A}} \widehat{v}_j(O_j)$. Therefore, the algorithm terminates only when we have the desired approximation to the welfare among agents in $\overline{A}$. This establishes the correctness of Algorithm 3.

For the run-time analysis, note that in every iteration of the while-loop in the algorithm the social welfare increases by at least $\frac{1}{225\sqrt{n}}$. Since the functions $\widehat{v}_j$s are upper bounded by $\frac{1}{\sqrt{n}}$, the maximum possible social welfare is $\sqrt{n}$. Hence, the while-loop iterates at most $225n$. Given that each iteration of the loop executes in polynomial time (using value, demand, and XOS oracle access to the valuations $v_j$s), we get that the algorithm computes an allocation in polynomial time. This establishes the theorem. ◀

## 5 Sublinear Approximation Algorithm for Nash Social Welfare

The section establishes our main result, the approximation ratio of Algorithm 1 for Nash social welfare, through a baroque case analysis. Recall that Algorithm 1 first removes $n \log n$ goods by taking repeated matchings. Then, the remaining goods are randomly partitioned into subsets R and R′. A discrete moving knife subroutine is executed over the goods in R (Algorithm 2) and Algorithm 3 partitions the goods in R′ to (approximately) maximize social welfare under the capped valuations $\widehat{v}_i$s.

▶ **Theorem 1** (Main Result.). *Given instance $\langle [n], [m], \{v_i\}_{i \in [n]} \rangle$, with XOS and demand oracle access to (monotone and XOS) valuations $v_i$s, Algorithm 1 computes (with high probability) an $O(n^{53/54})$ approximation to the optimal Nash social welfare.*

Recall that $\mathcal{Q} = (\mu(i) + \pi(i) + X_i + Y_i)_{i \in [n]}$ denotes the allocation returned by Algorithm 1; here we use notation as in Algorithm 1. Also, as before, $\mathcal{N} = (N_1, \ldots, N_n)$ denotes a Nash optimal allocation and $g_i^* = \arg\max_{g \in N_i} v_i(g)$ for all agents $i \in [n]$. For analytic purposes, we will consider the allocation in which the goods $g_i^*$ are included in the bundles $Q_i = \mu(i) + \pi(i) + X_i + Y_i$, in lieu of the goods $\mu(i)$; specifically, throughout this section write $Q_i^* \coloneqq g_i^* + \pi(i) + X_i + Y_i$, for all agents $i \in [n]$, and allocation $\mathcal{Q}^* \coloneqq (Q_1^*, \ldots, Q_n^*)$.

The next lemma (proof in the full version [5]) shows that the Nash social welfare of allocation $\mathcal{Q}$ is within a factor of $1/2$ of the Nash social welfare of the allocation $\mathcal{Q}^*$.

▶ **Lemma 11.** *Let $\mathcal{Q} = (\mu(i) + \pi(i) + X_i + Y_i)_{i \in [n]}$ denote the allocation computed by Algorithm 1 and write allocation $\mathcal{Q}^* = (Q_1^*, \ldots, Q_n^*)$ with bundles $Q_i^* \coloneqq g_i^* + \pi(i) + X_i + Y_i$, for all $i \in [n]$. Then, $\mathrm{NSW}(\mathcal{Q}) \geq \frac{1}{2}\mathrm{NSW}(\mathcal{Q}^*)$.*

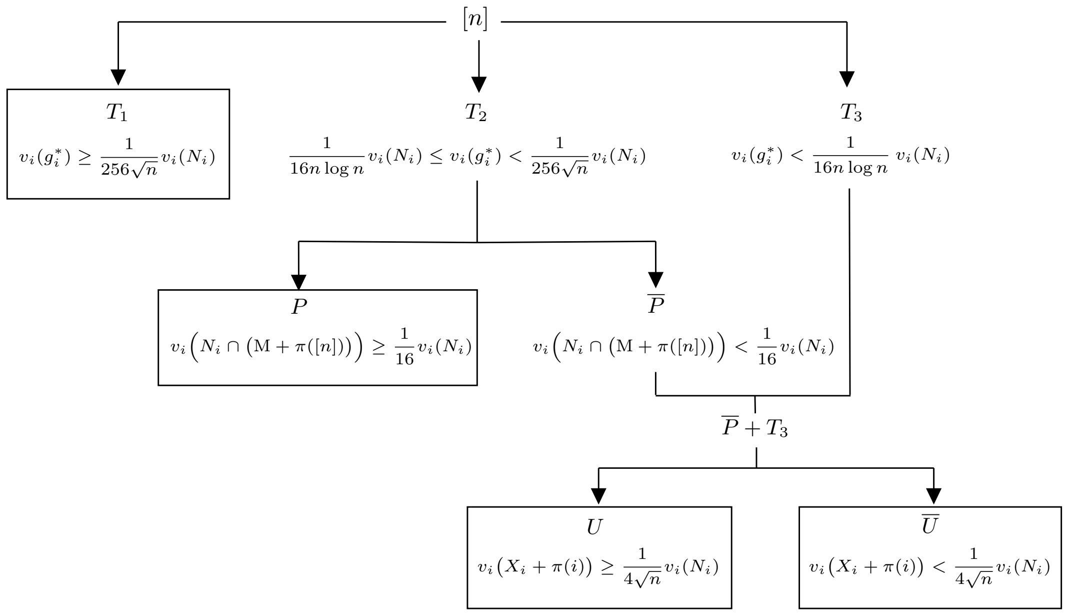


**Figure 1** The partitions (of the set of agents $[n]$) used in the analysis of Algorithm 1.

For a case analysis, we fist partition the agents into different types, $T_1$, $T_2$, and $T_3$, depending on the value they have for their $g_i^*$; see Figure 1. Specifically,

$$T_1 := \left\{ i \in [n] \ : \ v_i(g_i^*) \geq \frac{1}{256\sqrt{n}} v_i(N_i) \right\},$$
$$T_2 := \left\{ i \in [n] \ : \frac{1}{16n\log n} v_i(N_i) \leq v_i(g_i^*) < \frac{1}{256\sqrt{n}} v_i(N_i) \right\}, \quad \text{and}$$
$$T_3 := \left\{ i \in [n] \ : v_i(g_i^*) < \frac{1}{16n\log n} v_i(N_i) \right\}.$$

Note that agents in $T_1$ achieve an $O(\sqrt{n})$ approximation if they receive their optimal goods $g_i^*$s. To show that most other agents also get a sublinear approximation, we sub-divide the sets $T_2$ and $T_3$ based on the subsets computed in Algorithm 1.

In particular, we partition $T_2$ into two subsets: $P := \left\{ i \in T_2 \ : \ v_i\big(N_i \cap (\mathrm{M} + \pi([n]))\big) \geq \frac{1}{16} v_i(N_i) \right\}$ and $\overline{P} := \left\{ i \in T_2 \ : \ v_i\big(N_i \cap (\mathrm{M} + \pi([n]))\big) < \frac{1}{16} v_i(N_i) \right\}$. Note that the valuation $v_i$ is XOS (subadditive), hence, for all agents $i \in \overline{P}$ we have $v_i\big(N_i \setminus (\mathrm{M} + \pi([n]))\big) \geq \frac{15}{16} v_i(N_i)$.

It will also be helpful to consider the following partition of $\overline{P} + T_3$, based on the values obtained by $X_i$ (the bundle computed by the discrete moving knife procedure) and the matched good $\pi(i)$.

$$U := \left\{ i \in \overline{P} + T_3 \ : \ v_i(X_i + \pi(i)) \geq \frac{1}{4\sqrt{n}} v_i(N_i) \right\}.$$
$$\overline{U} := \left\{ i \in \overline{P} + T_3 \ : \ v_i(X_i + \pi(i)) < \frac{1}{4\sqrt{n}} v_i(N_i) \right\}.$$

The remainder of the section considers the following two exhaustive cases and shows that in both we achieve the desired approximation ratio of Algorithm 1.

**Case 1:** $|T_1 + P + U| \geq \frac{n}{27}$

**Case 2:** $|\overline{U}| \geq \frac{26n}{27}$

Specifically, in both cases, we show that the allocation $\mathcal{Q}^*$ (and consequently the computed allocation $\mathcal{Q}$) achieves a sublinear approximation to $\mathrm{NSW}(\mathcal{N})$. The rest of the proof is structured as follows:

- **Sublinear Approximation for agents in $T_1$, $P$, and $U$**: Lemmas 12, 13, and 14.
- **Linear Approximation for $T_3$ and $\overline{P}$**: Lemmas 15 and 16.
- **Sublinear Approximation Guarantee in Case 1**: Lemma 17.
- **Properties for Invoking Algorithm 3**: Propositions 18 and 19 build upon Lemmas 15 and 16. They show that, for Case 2, the properties required to apply CappedSocialWelfare subroutine hold.
- **Sublinear Approximation Guarantee in Case 2**: Lemma 20.
- Theorem 1 finally follows from Lemmas 11, 17 and 20.

We defer all the omitted proofs to the full version of the paper [5].

First for the agents in $T_1$, we have a sublinear approximation.

▶ **Lemma 12.** *For each agent $i \in T_1$ we have $v_i(Q_i^*) \geq \frac{1}{256\sqrt{n}} v_i(N_i)$.*

The next lemma addresses agents in the set $P = \left\{ i \in T_2 \ : \ v_i\left(N_i \cap (M + \pi([n]))\right) \geq \frac{1}{16} v_i(N_i) \right\}$.

▶ **Lemma 13.** *For agents in the set $P$ we have $\left(\prod_{i \in P} v_i(Q_i^*)\right)^{1/n} \geq \frac{1}{2} \left(\prod_{i \in P} \frac{1}{\sqrt{n}} v_i(N_i)\right)^{1/n}$.*

The following lemma addresses agents in the set $U = \{ i \in \overline{P} + T_3 \ : \ v_i(X_i + \pi(i)) \geq \frac{1}{4\sqrt{n}} v_i(N_i) \}$.

▶ **Lemma 14.** *For each agent $i \in U$ we have $v_i(Q_i^*) \geq \frac{1}{4\sqrt{n}} v_i(N_i)$.*

The following lemmas show that even if we restrict attention to the assignments made before the fourth phase in Algorithm 1 (i.e., if we consider $(X_i + \pi(i))$s), with high probability, each agent $i \in T_3 + \overline{P}$ achieves a linear approximation with respect to $v_i(N_i)$.

▶ **Lemma 15.** *For each agent $i \in T_3$, we have (with high probability) $v_i(X_i) \geq \frac{1}{64n} v_i(N_i)$, where $(X_1, \ldots, X_n)$ is the allocation returned by the DiscreteMovingKnife subroutine in Step 9 of Algorithm 1.*

As mentioned previously, for all agents $i \in \overline{P}$, we have $v_i\left(N_i \setminus (M + \pi([n]))\right) \geq \frac{15}{16} v_i(N_i)$. We prove the following lemma for the set $\overline{P}$.

▶ **Lemma 16.** *For every agent $i \in \overline{P}$, we have (with high probability) $v_i(X_i + \pi(i)) \geq \frac{1}{64n} v_i(N_i)$, where $(X_1, \ldots, X_n)$ is the allocation returned by the DiscreteMovingKnife subroutine in Step 9 of Algorithm 1 and $\pi(\cdot)$ is the matching computed in Step 6 of the algorithm.*

The sublinear approximation ratio[6] for Algorithm 1 under Case 1: $|T_1 + P + U| \geq \frac{n}{27}$ is established in the following lemma.

▶ **Lemma 17.** *If $|T_1 + P + U| \geq \frac{n}{27}$, then the Nash social welfare of allocation $\mathcal{Q}^*$ is at least $\frac{c'}{n^{53/54}}$ times the optimal Nash social welfare, $\mathrm{NSW}(\mathcal{Q}^*) \geq \frac{c'}{n^{53/54}} \mathrm{NSW}(\mathcal{N})$. Here, $c' \in \mathbb{R}_+$ is a fixed constant.*

[6] Recall that for the computed allocation $\mathcal{Q}$ we have $\mathrm{NSW}(\mathcal{Q}) \geq \frac{1}{2} \mathrm{NSW}(\mathcal{Q}^*)$ (Lemma 11).

Complementing the previous result, we now consider Case 2: $|\overline{U}| \geq \frac{26}{27}n$. Recall that to apply the guarantee obtained for CappedSocialWelfare (i.e., Theorem 7), we need the instance at hand to satisfy properties **P1** and **P2**, with some underlying allocation $\mathcal{O} = (O_1, \ldots, O_n)$ and set of agents $\overline{A} \subseteq [n]$. The following propositions show that these properties hold for the instance $\langle [n], \mathrm{R}', \{v_i\}_i \rangle$, bundles $O_j = N_j \cap \mathrm{R}'$ (for all agents $j \in [n]$) and subset $\overline{A} = \overline{U}$. This will enable us to instantiate Theorem 7 in the next subsection.

▶ **Proposition 18.** *If $|\overline{U}| \geq \frac{26n}{27}$, then (with high probability) the allocation, $\mathcal{O} = (O_1, \ldots, O_n)$, with bundles $O_i = N_i \cap \mathrm{R}'$ (for all $i \in [n]$), and subset $\overline{A} = \overline{U}$ satisfy property **P1** mentioned above.*

Next, we will address property **P2**.

▶ **Proposition 19.** *With high probability, for each agent $i \in \overline{U}$ and each good $g' \in N_j \cap \mathrm{R}'$, we have $\widehat{v}_i(g') < \frac{1}{2\sqrt{n}}$, i.e., property **P2** holds with bundles $O_j = N_j \cap \mathrm{R}'$ (for all agents $j \in [n]$) and subset $\overline{A} = \overline{U}$.*

Using these conditions, we can prove the sublinear approximation in Case 2.

▶ **Lemma 20.** *If $|\overline{U}| \geq \frac{26n}{27}$, then the Nash social welfare of allocation $\mathcal{Q}^*$ is at least $\frac{c}{n^{53/54}}$ times the optimal Nash social welfare,*

$$\mathrm{NSW}(\mathcal{Q}^*) \geq \frac{c}{n^{53/54}} \mathrm{NSW}(\mathcal{N}).$$

*Here, $c \in \mathbb{R}_+$ is a fixed constant.*

## 5.1 Proof of Theorem 1

We now restate Theorem 1 and show that it follows from Lemmas 17 and Lemma 20.

▶ **Theorem 1** (Main Result.). *Given instance $\langle [n], [m], \{v_i\}_{i \in [n]} \rangle$, with XOS and demand oracle access to (monotone and XOS) valuations $v_i$s, Algorithm 1 computes (with high probability) an $O(n^{53/54})$ approximation to the optimal Nash social welfare.*

**Proof.** Recall that $\mathcal{Q}$ is the allocation returned by Algorithm 1 and $\mathcal{N}$ is a Nash optimal allocation. To prove the theorem, we consider the (previously-mentioned) exhaustive cases:

**Case 1:** $|T_1 + U + P| \geq \frac{n}{27}$. In this case, Lemmas 11 and 17 give us $\mathrm{NSW}(\mathcal{Q}) \geq \frac{1}{2}\mathrm{NSW}(\mathcal{Q}^*) \geq \frac{c'}{2n^{53/54}}\mathrm{NSW}(\mathcal{N})$.

**Case 2:** $|\overline{U}| \geq \frac{26n}{27}$. Here, via Lemmas 11 and 20, we obtain (with high probability) $\mathrm{NSW}(\mathcal{Q}) \geq \frac{1}{2}\mathrm{NSW}(\mathcal{Q}^*) \geq \frac{c}{2n^{53/54}}\mathrm{NSW}(\mathcal{N})$.

This case analysis shows that overall Algorithm 1 achieves an approximation ratio of $O(n^{53/54})$ for the Nash social welfare maximization problem, under XOS valuations. The theorem stands proved. ◀

▶ Remark. Here we address the corner case wherein for some agents $z \in [n]$ the value of the good assigned in Step 6 of Algorithm 1 is zero, i.e., $v_z(\pi(z)) = 0$. Note that for remaining agents $i$, we have $v_i(\pi(i)) > 0$ and, hence, $v_i(X_i + \pi(i)) > 0$. Consequently, for such agents $i$, the parameter $\beta_i$ (considered in Step 10 of the algorithm) is well-defined.

Note that, if for an agent $z$ we have $v_z(\pi(z)) = 0$, then $z$ necessarily belongs to either set $T_1$ or set $P$. This follows from the observation that, for such an agent $z$, all the goods in the set $[m] \setminus \big(\mathrm{M} + \pi([n])\big)$ are of zero value. Equivalently, all the goods of nonzero value for $z$ are contained in $\big(\mathrm{M} + \pi([n])\big)$. Therefore, $z$ has nonzero value for at most $n \log n + n \leq 2n \log n$ goods. For an XOS (subadditive) valuation $v_z$, this implies that $v_z(g_z^*)$ cannot be less than $\frac{1}{16n \log n} v_z(N_z)$ and, hence, $z \notin T_3$. Furthermore, the fact that $v_z\left([m] \setminus \big(\mathrm{M} + \pi([n])\big)\right) = 0$ gives us $z \notin \overline{P}$. Therefore, $z$ must be contained in $T_1 \cup P$.

We exclude such agents $z$ from phases three and four of Algorithm 1; this ensures that we do not have to consider $\beta_z$. Then, such agents $z$ are directly considered in Step 12 with $X_z = Y_z = \emptyset$. Since $z \in T_1 \cup P$, the arguments from Lemmas 12 and 13 provide a sublinear guarantee for $z$ even with $X_z = Y_z = \emptyset$.

For the remaining agents $i$ (with the property that $v_i(\pi(i)) > 0$), the guarantees obtained for phases three and four (in particular, the ones obtained in Lemmas 15 and 16) in fact improve, since the number of agents under consideration gets reduced. These observations imply that the sublinear approximation guarantee holds as is in Case 1. For Case 2 (i.e., when $|\overline{U}| > {}^{26n}/_{27}$), note that, $\overline{U} \cap (T_1 \cup P) = \emptyset$. Therefore, Lemma 20 is applicable and we obtain the stated approximation ratio throughout.

## 6 Conclusion and Future Work

This work breaks the $O(n)$ approximation barrier that holds for NSW maximization under XOS valuations in the value-oracle model. In particular, using demand and XOS oracles, we obtain the first sublinear approximation algorithm for maximizing NSW with XOS valuations. A key innovative contribution of the work is the connection established between NSW and capped social welfare. This connection builds upon the following high-level idea: to achieve a sublinear approximation for NSW, it suffices first to obtain an $O(n)$-approximation guarantee and, then, ensure that a constant fraction of the agents additionally achieve a sublinear approximation of their optimal valuation.

Understanding the limitations of demand queries – in the NSW context – is a relevant direction for future work. While we rule out a $(1 - 1/e)$-approximation with subexponentially many queries and it is known that NSW maximization is APX-hard under demand queries (see, e.g., [8]), it would be interesting to obtain stronger inapproximability results. Developing a sublinear approximation guarantee for subadditive valuations will also be interesting.

## References


1. Nima Anari, Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. Nash social welfare, matrix permanent, and stable polynomials. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67, page 36. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017.
2. Nima Anari, Tung Mai, Shayan Oveis Gharan, and Vijay V Vazirani. Nash social welfare for indivisible items under separable, piecewise-linear concave utilities. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2274–2290. SIAM, 2018.
3. Moshe Babaioff, Tomer Ezra, and Uriel Feige. Fair and truthful mechanisms for dichotomous valuations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:5119–5126, May 2021. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/16647`.
4. Siddharth Barman, Umang Bhaskar, Anand Krishna, and Ranjani G. Sundaram. Tight approximation algorithms for p-mean welfare under subadditive valuations. In *ESA*, 2020.
5. Siddharth Barman, Anand Krishna, Pooja Kulkarni, and Shivika Narang. Sublinear approximation algorithm for Nash social welfare with XOS valuations. *arXiv preprint arXiv:2110.00767*, 2021.

6 Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Finding fair and efficient allocations. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 557–574, 2018.

7 Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Greedy algorithms for maximizing nash social welfare. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 7–13, 2018.

8 Siddharth Barman and Paritosh Verma. Approximating nash social welfare under binary xos and binary subadditive valuations. *To appear at WINE*, 2021.

9 Steven J Brams, Steven John Brams, and Alan D Taylor. *Fair Division: From cake-cutting to dispute resolution*. Cambridge University Press, 1996.

10 Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. *Handbook of computational social choice*. Cambridge University Press, 2016.

11 Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, 7(3):1–32, 2019.

12 Bhaskar Ray Chaudhury, Jugal Garg, and Ruta Mehta. Fair and efficient allocations under subadditive valuations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5269–5276, 2021.

13 Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay V Vazirani, and Sadra Yazdanbod. Convex program duality, fisher markets, and nash social welfare. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 459–460, 2017.

14 Richard Cole and Vasilis Gkatzelis. Approximating the nash social welfare with indivisible items. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 371–380, 2015.

15 Andreas Darmann and Joachim Schauer. Maximizing nash product social welfare in allocating indivisible goods. *European Journal of Operational Research*, 247(2):548–559, 2015.

16 Shahar Dobzinski, Noam Nisan, and Michael Schapira. Approximation algorithms for combinatorial auctions with complement-free bidders. *Mathematics of Operations Research*, 35(1):1–13, 2010.

17 Shahar Dobzinski and Michael Schapira. An improved approximation algorithm for combinatorial auctions with submodular bidders. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1064–1073, 2006.

18 Uriel Feige. On maximizing welfare when utility functions are subadditive. *SIAM Journal on Computing*, 39(1):122–142, 2009.

19 Jugal Garg, Martin Hoefer, and Kurt Mehlhorn. Approximating the nash social welfare with budget-additive valuations. *CoRR*, abs/1707.04428, 2017. `arXiv:1707.04428`.

20 Jugal Garg, Martin Hoefer, and Kurt Mehlhorn. Approximating the nash social welfare with budget-additive valuations. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2326–2340. SIAM, 2018.

21 Jugal Garg, Edin Husić, and László A. Végh. Approximating nash social welfare under rado valuations. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, pages 1412–1425. Association for Computing Machinery, 2021.

22 Jugal Garg, Pooja Kulkarni, and Rucha Kulkarni. Approximating nash social welfare under submodular valuations through (un) matchings. In *Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms*, pages 2673–2687. SIAM, 2020.

23 Mamoru Kaneko. An extension of the nash bargaining problem and the nash social welfare function. *Theory and Decision*, 12(2):135–148, 1980.

24 Mamoru Kaneko and Kenjiro Nakamura. The nash social welfare function. *Econometrica: Journal of the Econometric Society*, pages 423–435, 1979.

25 Euiwoong Lee. Apx-hardness of maximizing nash social welfare with indivisible items. *Information Processing Letters*, 122:17–20, 2017.

**26** Wenzheng Li and Jan Vondrák. A constant-factor approximation algorithm for nash social welfare with submodular valuations. *To appear at FOCS*, 2021.

**27** Herve Moulin. *Fair Division and Collective Welfare*. MIT Press, 2003.

**28** Trung Thanh Nguyen and Jörg Rothe. Minimizing envy and maximizing average nash social welfare in the allocation of indivisible goods. *Discrete Applied Mathematics*, 179:54–68, 2014.

**29** Noam Nisan, T Roughgarden, E Tardos, and Vijay V Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.