On the Fair Allocation to Asymmetric Agents with Binary XOS Valuations

Ziheng Chen*
State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences
School of Computer Science and Technology, University of Chinese Academy of Sciences
Beijing, China
chenzihang15@mails.ucas.ac.cn

Bo Li*
Department of Computing
The Hong Kong Polytechnic University
Hong Kong, China
comp-bo.li@polyu.edu.hk

Zihan Luo*
State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences
School of Computer Science and Technology, University of Chinese Academy of Sciences
Beijing, China
luozihan23s@ict.ac.cn

Jialin Zhang*†
Institute of Computing Technology, Chinese Academy of Sciences
University of Chinese Academy of Sciences
Beijing, China
zhangjialin@ict.ac.cn

ABSTRACT

We study the problem of allocating m indivisible goods among n agents, where each agent's valuation is fractionally subadditive (XOS). With respect to AnyPrice Share (APS) fairness, Kulkarni et al. [22] showed that, when agents have binary marginal values, a 0.1222-APS allocation can be found in polynomial time, and there exists an instance where no allocation is better than 0.5-approximate APS. Very recently, Feige and Grinberg [12] extended the problem to the asymmetric case, where agents may have different entitlements, and improved the approximation ratio to $1/6$ for general XOS valuations. In this work, we focus on the asymmetric setting with binary XOS valuations, and further improve the approximation ratio to $1/2$, which matches the known upper bound. We also present a polynomial-time algorithm to compute such an allocation. Beyond APS fairness, we also study the weighted maximin share (WMMS) fairness. Farhadi et al. [10] showed that, a $1/n$-WMMS allocation always exists for agents with general additive valuations, and that this approximation ratio is tight. We extend this result to general XOS valuations, where a $1/n$-WMMS allocation still exists, and this approximation ratio cannot be improved even when marginal values are binary. This shows a sharp contrast to binary additive valuations, where an exact WMMS allocation exists and can be found in polynomial time.

KEYWORDS

fair allocation; XOS; APS; WMMS

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/BSDW5494

ACM Reference Format:
Ziheng Chen, Bo Li[1], Zihan Luo[1], and Jialin Zhang[1]. 2026. On the Fair Allocation to Asymmetric Agents with Binary XOS Valuations. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/BSDW5494

1 INTRODUCTION

In this paper, we study the fair allocation of m indivisible goods among n agents. A central and widely accepted fairness criterion in this problem is the maximin share (MMS), introduced by Budish [9]. However, prior work has shown that there exist instances where no allocation can guarantee MMS fairness [15, 23, 27]. This raises a fundamental question in fair allocation: to what extent can MMS fairness be guaranteed to be satisfiable? This question has inspired a substantial body of work, and for additive, submodular, and XOS valuation functions, constant-factor approximations to MMS fairness have been established; see, for example, [1, 3, 31]. Babaioff et al. [6] proposed an alternative fairness notion, AnyPrice share (APS), and proved the existence of a constant-approximate APS allocation under additive valuations, which was later generalized to submodular valuations [31].

In addition to research on general classes of valuations, significant attention has been given to valuations with binary marginals, such as matroid-rank valuations (i.e., submodular valuations with binary marginals). On one hand, these binary functions are well-suited for modeling real-world scenarios – such as the fair allocation of public housing units [8] – and make it easier for agents to express their preferences. On the other hand, they allow for stronger fairness guarantees. For instance, under matroid-rank valuations, an exact MMS allocation is always guaranteed to exist and can be efficiently computed [8, 32]. Furthermore, Kulkarni et al. [22] showed that in this setting, APS is exactly equal to MMS, ensuring that exact APS allocations are always attainable. This naturally leads to the question: what about binary XOS functions, which generalize binary submodular functions within the complement-free

---

*These authors contributed equally to this work.
†Corresponding author.

hierarchy? Kulkarni et al. [22] showed that, in this case, MMS and APS may not be the same, but they satisfy MMS $\leq$ APS $\leq$ 2MMS+1. Given that the algorithm by Li and Vetta [26] guarantees a 0.366-approximate MMS allocation, this relationship immediately implies a 0.122-approximate APS allocation. Unfortunately, Kulkarni et al. [22] also showed that it is impossible to achieve an approximation better than 0.5 for APS under binary XOS valuations.

Very recently, Feige and Grinberg [12] improved upon previous results by showing that a $\frac{4}{17}(\approx 0.235)$-approximate APS allocation exists under the general XOS valuations. They also extended their analysis to a more general setting where agents are asymmetric—that is, agents have different entitlements to the goods, with those holding higher entitlements expected to receive a larger share. This asymmetric setting is particularly relevant to real-world applications. As discussed in [10], in many religions, cultures, and legal systems, inherited wealth is often distributed unequally, and the allocation of natural resources between neighboring countries is frequently based on geographic, economic, or political considerations. Feige and Grinberg [12] proved that, for general XOS valuations in the asymmetric setting, a $\frac{1}{6}$-approximate APS allocation is guaranteed to exist.

While the results of Feige and Grinberg [12] have significantly advanced our understanding of approximate MMS and APS fairness under XOS valuations, several open questions from [22] remain unresolved. First, for binary XOS valuations, the optimal approximation ratio is still unknown, which is known to be between $\frac{4}{17}$ and $\frac{1}{2}$ for the symmetric setting, and between $\frac{1}{6}$ and $\frac{1}{2}$ for the asymmetric setting. Second, in the asymmetric setting, the relationship between APS and weighted MMS (WMMS, as defined in [10]), is still unknown, and the optimal approximation ratio of WMMS is unknown. In this paper, we answer the above two questions.

1.1 Our Contribution

In this paper, we build upon the work of [22] to investigate the fair allocation of m indivisible goods among n agents, where agents' valuations are binary XOS. We focus on APS and WMMS fairness, and the general setting when agents have different entitlements. The main results and their comparisons to the related work are summarized in Table 1.

Our contributions are twofold. First, for APS fairness, we show that a $\frac{1}{2}$-approximate APS allocation always exists, and thus this approximation ratio is tight. Our result applies to the general asymmetric case; see Theorem 1. In the symmetric setting, since APS is always at least as large as MMS, our result also guarantees the existence of a $\frac{1}{2}$-approximate MMS allocation. Furthermore, we provide a polynomial-time implementation to compute such an allocation; see Theorem 2.

In the asymmetric setting, Farhadi et al. [10] first introduced WMMS fairness to account for agents with different weights. In this context, APS is no longer an upper bound for WMMS; indeed, there are instances (with agents having general additive valuations) where $\frac{\mathsf{APS}}{\mathsf{WMMS}}$ is infinitesimal for some agent; see Proposition 1. Farhadi et al. [10] further demonstrated that a round-robin algorithm guarantees a $\frac{1}{n}$-approximate WMMS allocation for additive valuations. Extending beyond additive valuations, we generalize the result of Farhadi et al. [10] by proving that a $\frac{1}{n}$-approximate WMMS allocation always exists for arbitrary XOS valuations (Theorem 3). Nevertheless, we show that for XOS valuations – even when restricted to binary cases – a better than $\frac{1}{n}$-approximate WMMS allocation cannot be guaranteed (Theorem 4). In contrast, we prove that an exact WMMS allocation is achievable for binary additive valuations, underscoring the inherent challenges of attaining WMMS fairness for binary XOS valuations (Theorem 5).

1.2 Related Work

MMS fairness was first introduced in Budish [9] as a relaxation of proportionality [30]. However, there are instances for which no MMS allocation exists [15, 23, 27]. For additive valuations, Kurokawa et al. [23] proved the existence of $\frac{2}{3}$-MMS allocations. Subsequently, Ghodsi et al. [17] improved the approximation ratio to $\frac{3}{4}$, which was further improved to $\frac{3}{4} + o(1)$ by Akrami et al. [2], Garg and Taki [16], $\frac{3}{4} + \frac{3}{3836}$ by Akrami and Garg [1], and most recently to $\frac{10}{13}$ by Heidari et al. [18]. For submodular valuations, Barman and Krishnamurthy [7] and Ghodsi et al. [17], respectively, designed algorithms to compute 0.21-MMS and $\frac{1}{3}$-MMS allocations. Ghodsi et al. [17] showed that a better than $\frac{3}{4}$-MMS allocation may not exist. Recently, Uziahu and Feige [31] improved the approximation ratio to $\frac{10}{27}$. For XOS valuations, Ghodsi et al. [17] proved that the best possible approximation ratio is between $\frac{1}{5}$ and $\frac{1}{2}$. Later, Akrami et al. [3] designed an algorithm computing 0.23-MMS allocations. For general subadditive valuations, approximate MMS fair algorithms are given in [11, 13, 17, 28, 29]. Finally, when agents have asymmetric entitlements, Farhadi et al. [10] introduced weighted MMS fairness, and proved that the best possible approximation ratio for additive valuations is $\frac{1}{n}$.

Babaioff et al. [6] introduced APS fairness, which addresses some of the modeling concerns of MMS, especially when agents have different weights. They designed a 0.667-APS algorithm for additive valuations. Later, Uziahu and Feige [31] extended the setting to submodular valuations and presented a $\frac{1}{3}$-approximation algorithm. Recently, Feige and Grinberg [12] showed the existence of $\frac{1}{6}$-APS allocations for XOS valuations, and improved the approximation ratio to $\frac{4}{17}$-APS allocation when agents have equal entitlements.

Although all the above work considers the allocation of goods, the minor problem of chores has also been widely studied. For example, with equal entitlements, the MMS fairness is studied in [5, 19, 20] for additive valuations and in [25] for more general valuations. With asymmetric entitlements, the WMMS fairness is studied in [4, 33], and APS fairness is studied in [6, 14, 24].

2 PRELIMINARIES

We study the problem of fairly allocating a set of m indivisible goods among n agents. Let $[k] = \{1, 2, \ldots, k\}$ for a positive integer k. Denote the set of goods by $M := \{g_1, g_2, \ldots, g_m\}$ and the set of agents by $N := \{a_1, a_2, \ldots, a_n\}$. The preference of each agent $a_i \in N$ is defined by a valuation function $v_i : 2^M \to \mathbb{R}_{\geq 0}$ over the set of goods. Specifically, $v_i(S)$ is the value that a_i has for the subset of goods $S \subseteq M$. Furthermore, we emphasize the imbalance between agents with their entitlements $(b_i)_{i\in[n]}$. Therefore, we represent a fair allocation instance by the quadruple $(N, M, (v_i)_{i\in n}, (b_i)_{i\in[n]})$.

An allocation, $A := (A_1, \ldots, A_n)$, is a partition of all the goods among the n agents, i.e., $A_i \cap A_j = \emptyset$ for all $i \neq j$ and $\bigcup_{i\in[n]} A_i = M$.

Setting	APS		WMMS	
	Symmetric	Asymmetric	Symmetric	Asymmetric
Additive	$\frac{3}{5}$ (Babaioff et al. [6])	$\frac{3}{5}$ (Babaioff et al. [6])	$\frac{10}{13}$ (Heidari et al. [18])	$\frac{1}{n}$ (Farhadi et al. [10])
Binary XOS	$0.1222 \xrightarrow{\textbf{Theorem 2}} \frac{1}{2}$ (Kulkarni et al. [22])	$\frac{1}{2}$ (**Theorem 2**)	$0.3666 \xrightarrow[\text{Hummel [21]}]{\textbf{Corollary 1}} \frac{1}{2}$ (Kulkarni et al. [22])	$\frac{1}{n}$ (**Theorem 3**)
General XOS	$\frac{4}{17}$ (Feige and Grinberg [12])	$\frac{1}{6}$ (Feige and Grinberg [12])	$\frac{4}{17}$ (Feige and Grinberg [12])	$\frac{1}{n}$ (**Theorem 3**)

Table 1: Main results. We remark that all our approximations are tight. For clarity, we provide a comparison with the best-known approximations in related settings.

We also define a partial allocation, denoted by $P = (P_1, \ldots, P_n)$, as a partition of any subset of goods, where $P_i \cap P_j = \emptyset$ for all $i \neq j$ and $\bigcup_{i\in[n]} P_i \subseteq M$.

2.1 Fairness Notions

For the sake of simplicity, in a fair allocation instance $(N, M, (v_i)_{i\in n}, (b_i)_{i\in[n]})$, let

$$\Pi_n = \{S = (S_i)_{i\in[n]} \mid M = \cup_{i\in[n]} S_i, S_i \cap S_j = \emptyset, i \neq j\}$$

be the set of all partitions of M to n bundles.

The maximin share is an analogy of the cake-cutting problem, where the cutter could only expect to obtain the minimum share of the cake. Therefore she need to find the allocation that would maximize the minimum share. In the unweighted version of maximin share, we treat all agents equally. Namely, each agent has the same entitlement $\frac{1}{n}$.

Definition 1 (Maximin Share). *For a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (\frac{1}{n})_{i\in[n]})$, the maximin share (MMS) of a_i is*

$$\mathsf{MMS}_i^N(M) = \max_{S\in\Pi_n} \min_{j\in[n]} v_i(S_j).$$

As an extension, the weighted maximin share takes the entitlements into consideration. Now the cutter indeed proposes an allocation according to her valuation function and the agents' entitlements instead of just a partition.

Definition 2 (Weighted Maximin Share). *For a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$, the weighted maximin share (WMMS) of a_i is*

$$\mathsf{WMMS}_i^N(M) = \max_{S\in\Pi_n} \min_{j\in[n]} v_i(S_j) \frac{b_i}{b_j}.$$

It is obvious that when all entitlements are equal, the weighted maximin share degenerates to the maximin share. Furthermore, we call the partition that reaches the WMMS value the WMMS partition.

Definition 3 (WMMS Partition). *For a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$, a WMMS partition of a_i is*

$$S^i \in \arg\max_{S\in\Pi_n} \min_{j\in[n]} v_i(S_j^i) \frac{b_i}{b_j},$$

where S_j^i is the bundle a_i wants to assign to a_j.

Informally, AnyPrice Share indicates the value that an agent a_i can guarantee herself at any given price of items, based on her entitlement b_i.

Definition 4 (AnyPrice Share). *Let $\mathcal{P}$ represent the price vector simplex corresponding to the set of goods M, or formally*

$$\mathcal{P} = \left\{ p = (p_1, p_2, \ldots, p_m) \geq 0 \;\Big|\; \sum_{i\in[m]} p_i = 1 \right\}.$$

For a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$, the APS value of a_i is

$$\mathsf{APS}_i^N(M) := \min_{p\in\mathcal{P}} \max_{S\subseteq M, p(S)\leq b_i} v_i(S),$$

where $p(S)$ is the sum of prices of goods in S.

For brevity we will refer to $(\mathsf{W})\mathsf{MMS}_i^N(M)$ as $(\mathsf{W})\mathsf{MMS}_i$ and $\mathsf{APS}_i^N(M)$ as APS_i if the instance is clear by the context.

We emphasize that in the definition of (weighted) maximin share and AnyPrice share of some a_i, only her own value function counts. The other agents' participation is limited to their entitlements.

An allocation A is α-APS (similarly, α-(W)MMS) if $v_i(A_i) \geq \alpha\mathsf{APS}_i$ (similarly, $v_i(A_i) \geq \alpha(\mathsf{W})\mathsf{MMS}_i$) for every agent $i \in N$.

2.2 Valuation Functions

This paper focuses on XOS valuations with binary marginals (or binary XOS valuations).

Definition 5 (Additive Function). *A function $v : 2^M \to \mathbb{R}_{\geq 0}$ is additive if the value of a set of goods is equal to the sum of the values of the goods in the set, that is, $v(S) = \sum_{g\in S} v(\{g\})$.*

Definition 6 (XOS Function). *A function $v : 2^M \to \mathbb{R}_{\geq 0}$ is XOS, or fractionally subadditive, iff there exists a collection of additive functions $\{l_t\}_{t\in[L]}$ such that, for every subset $S \subseteq M$, we have $v(S) = \max_{t\in[L]} l_t(S)$. Notice that the number of additive functions, $|L|$, can be exponentially large in m.*

Given an XOS function v, we suppose there is a query oracle O that receives a set S as input and computes $v(S)$ in time $O(1)$.

Furthermore, a valuation v has binary marginals if and only if $v(S \cup \{g\}) - v(S) \in \{0, 1\}$ for any subset of goods $S \subseteq M$ and any goods $g \in M$. As a direct consequence, the valuations we consider are monotonic: $v(S) \le v(T)$ for any subsets $S \subseteq T \subseteq M$. In addition, we require the valuations to satisfy $v_i(\emptyset) = 0$ for each $i \in [n]$.

When valuation v has binary marginals, a set of goods $S \subseteq M$ is said to be non-wasteful respect to valuation v, iff $v(S) = |S|$. Barman and Verma [8] proved that for any binary XOS function v and every set $S \subseteq M$, there exists a non-wasteful subset $X \subseteq S$ with the property that $v(X) = |X| = v(S)$, and the non-wasteful set X can be computed efficiently in time $O(m)$.

3 APS WITH BINARY XOS VALUATIONS

In this section, we focus on computing approximately APS allocations under binary XOS valuations. We first prove the existence of a $\frac{1}{2}$-APS allocation in Section 3.1, and then give a polynomial time implementation in Section 3.2. Combining with the upper bound in [22], our result is tight. Since APS is upper-bounded by MMS when agents have equal entitlements, our result implies a polynomial-time algorithm computing $\frac{1}{2}$-MMS allocations in the symmetric setting.

3.1 The Existence of $\frac{1}{2}$-APS Allocations

Our proof is constructive, and the algorithm is formally described in Algorithm 1. The algorithm returns a $\frac{1}{2}$-APS allocation if the APS values of all the agents are known. To make it a subprocess of our polynomial-time algorithm 2, we let Algorithm 1 output an unsatisfied agent if the input guessing values of APS' are too large.

Suppose the APS' of the agents are $s_1, s_2, \ldots, s_n$, then we can rename the agents such that

$$\frac{\lceil \frac{1}{2} s_1 \rceil}{b_1} \le \frac{\lceil \frac{1}{2} s_2 \rceil}{b_2} \le \cdots \le \frac{\lceil \frac{1}{2} s_n \rceil}{b_n}. \tag{1}$$

We prove that we can allocate each agent a non-wasteful bundle with value at least half of their APS.

Lemma 1. *For some agent a_i in an instance $(N, M, (v_i)_{i \in [n]}, (b_i)_{i \in [n]})$ with binary XOS valuations, suppose $S \subseteq M$ is a subset of goods, then there exists a non-wasteful bundle $B \subseteq M \setminus S$, satisfying $v_i(B) = |B| = \mathsf{APS}_i - \lfloor b_i |S| \rfloor$.*

Proof. If $S = \emptyset$, the lemma is straightforward by the definition of APS. Otherwise, consider the following prices for all goods l,

$$p_l = \begin{cases} \frac{1}{|S|}, & l \in S, \\ 0, & l \notin S. \end{cases}$$

With her entitlement b_i, a_i could at most afford $\lfloor b_i |S| \rfloor$ goods in S. While by the definition of APS, a_i could always afford at least APS_i value of items in M with any price. Therefore, for remaining goods, we have $v_i(M \setminus S) \ge \mathsf{APS}_i - \lfloor b_i |S| \rfloor$. Then by Barman and Verma [8], we could strip some goods off $M \setminus S$ to get the demanded non-wasteful bundle B, where $v_i(B) = |B| = \mathsf{APS}_i - \lfloor b_i |S| \rfloor$. □

Theorem 1. *If $s_i = \mathsf{APS}_i$ for $i \in [n]$, Algorithm 1 computes a $\frac{1}{2}$-APS allocation for any instance $(N, M, (v_i)_{i \in [n]}, (b_i)_{i \in [n]})$, where $(v_i)_{i \in [n]}$ are binary XOS valuations.*

Algorithm 1 $\frac{1}{2}$-APS Existence

```
Require: An instance $(N, M, (v_i)_{i \in [n]}, (b_i)_{i \in [n]})$ with binary XOS
         valuations, where the APS values $s_1, s_2, \ldots, s_n$ satisfy
         Inequality 1.
Ensure: If there exists a $\frac{1}{2}$-APS allocation, return this allocation
        $A = (A_1, A_2, \ldots, A_n)$. Otherwise, return an unsatisfied
        agent $a_i$.
1: for $i \leftarrow 1, 2, \ldots, n$ do
2:     if $v_i(M) \ge \lceil \frac{1}{2} s_i \rceil$ then
3:         Assign $a_i$ a non-wasteful bundle $A_i$ from $M$, where
           $v_i(A_i) = |A_i| = \lceil \frac{1}{2} s_i \rceil$.
4:         Set $M \leftarrow M \setminus A_i$.
5:     else
6:         return $a_i$
7: if $i = n$ and $M \ne \emptyset$ then
8:     Allocate the remaining items arbitrarily.
9:     return $A = (A_1, A_2, \ldots, A_n)$.
```

Proof. In Algorithm 1 we claimed the existence of $(A_i)_{i \in [n]}$ in line 3 when $s_i = \mathsf{APS}_i$ for $i \in [n]$. Now we prove by induction. For a_1, it is obvious that there is a bundle B_1 where $v_1(B_1) = |B_1| = \mathsf{APS}_1 \ge \lceil \frac{1}{2} \mathsf{APS}_1 \rceil$. Then similar to the proof of lemma 1, by Barman and Verma [8], we can find a non-wasteful bundle $A_1 \subseteq B_1$ such that $v_1(A_1) = |A_1| = \lceil \frac{1}{2} \mathsf{APS}_1 \rceil$.

We then assume that in the first k iterations where $1 \le k \le n-1$, $(a_i)_{i \in [k]}$ have received their bundles $(A_i)_{i \in [k]}$ satisfying $v_i(A_i) = |A_i| = \lceil \frac{1}{2} \mathsf{APS}_i \rceil$. We will prove that a_{k+1} could obtain a bundle A_{k+1} where $v_{k+1}(A_{k+1}) = |A_{k+1}| = \lceil \frac{1}{2} \mathsf{APS}_{k+1} \rceil$.

Let $S_{k+1} = \cup_{i=1}^{k} A_i$ be the set of items that has been taken by the first k agents. Substitute S to S_{k+1} in lemma 1, we know that $v_{k+1}(M \setminus S_{k+1}) \ge \mathsf{APS}_{k+1} - \lfloor b_{k+1} |S_{k+1}| \rfloor$. Therefore, if $\mathsf{APS}_{k+1} - \lfloor b_{k+1} |S_{k+1}| \rfloor \ge \lceil \frac{1}{2} \mathsf{APS}_{k+1} \rceil$, a_{k+1} could receive a bundle A_{k+1} as demanded.

$$\begin{aligned}
&\mathsf{APS}_{k+1} - \lfloor b_{k+1} |S_{k+1}| \rfloor = \mathsf{APS}_{k+1} - \left\lfloor b_{k+1} \sum_{i=1}^{k} |A_i| \right\rfloor \\
=&\mathsf{APS}_{k+1} - \left\lfloor b_{k+1} \sum_{i=1}^{k} \left\lceil \frac{1}{2} \mathsf{APS}_i \right\rceil \right\rfloor = \mathsf{APS}_{k+1} - \left\lfloor b_{k+1} \sum_{i=1}^{k} \frac{\lceil \frac{1}{2} \mathsf{APS}_i \rceil}{b_i} b_i \right\rfloor \\
\ge&\mathsf{APS}_{k+1} - \left\lfloor b_{k+1} \sum_{i=1}^{k} \frac{\lceil \frac{1}{2} \mathsf{APS}_{k+1} \rceil}{b_{k+1}} b_i \right\rfloor && (2) \\
=&\mathsf{APS}_{k+1} - \left\lfloor \left(\sum_{i=1}^{k} b_i \right) \left\lceil \frac{1}{2} \mathsf{APS}_{k+1} \right\rceil \right\rfloor \ge \mathsf{APS}_{k+1} - \left\lfloor \frac{1}{2} \mathsf{APS}_{k+1} \right\rfloor && (3) \\
=&\left\lceil \frac{1}{2} \mathsf{APS}_{k+1} \right\rceil.
\end{aligned}$$

The inequality 2 holds since agents are sorted in ascending order by $\frac{\lceil \frac{1}{2} \mathsf{APS}_i \rceil}{b_i}$ for $1 \le i \le n$. The inequality 3 holds since $k \le n-1$ and therefore $\sum_{i=1}^{k} b_i < 1$.

By induction, we have proved that each $a_i \in N$ can take a non-wasteful bundle A_i where $v_i(A_i) = |A_i| = \lceil \frac{1}{2} \mathsf{APS}_i \rceil$. □

3.2 A Polynomial-time Algorithm

Algorithm 1 assumes that we are given the APS values of all agents. However, it is unclear whether APS values can be computed in polynomial time. In the following, we provide a polynomial-time implementation, as described in Algorithm 2. Intuitively, Algorithm 2 iteratively guesses the APS values, beginning with their trivial upper bounds $s_i = \lfloor b_i m \rfloor$, and invokes Algorithm 1 to check if a desired allocation is returned. If not, an unsatisfied agent a_i is returned, indicating that the current guess for a_i's APS value is too high. Consequently, s_i is decreased, and the process continues in the next iteration.

Algorithm 2 $\frac{1}{2}$-APS Polynomial-Time Algorithm

```
Require: An instance with binary XOS valuations
         $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$
Ensure: An $\frac{1}{2}$-APS allocation $A = (A_1, A_2, \dots, A_n)$
 1: Set $M_0 \leftarrow M$.
 2: for $i \leftarrow 1, 2, \dots, n$ do
 3:     Initialize $s_i = \lfloor b_i m \rfloor$.        ▷ Upper bounds of APS
 4: while $A = (A_1, A_2, \dots, A_n)$ have not been found do
 5:     Rename the agents so that
        $\frac{\lceil \frac{1}{2}s_1 \rceil}{b_1} \le \frac{\lceil \frac{1}{2}s_2 \rceil}{b_2} \le \cdots \le \frac{\lceil \frac{1}{2}s_n \rceil}{b_n}$.
 6:     Execute Algorithm 1 with input $s_1, s_2, \dots, s_n$.
 7:     if Algorithm 1 returns an unsatisfied agent $a_i$ then
                                        ▷ Some $s_i$ is still too large.
 8:         Set $s_i \leftarrow s_i - 1$.
 9:         Reset $M \leftarrow M_0$.
10:     else
11:         return $A = (A_1, A_2, \dots, A_n)$
```

Lemma 2. *The time complexity of Algorithm 2 is $O(mn(m+\log n))$.*

Proof. Each time the while loop in line 4 is executed, either a $\frac{1}{2}$-APS allocation is found, or the guessing value s_i for some agent a_i is decreased by 1. The sum of the initial guessing APS values for all agents is $\sum_{i=1}^{n} s_i = \sum_{i=1}^{n} \lfloor b_i m \rfloor \le m$, so the while loop in line 4 can be executed at most m times.

During each iteration of the while loop, Algorithm 1 executes once, since it needs to find at most n times of non-wasteful bundles. If Algorithm 1 failed to find enough bundles, Algorithm 2 will modify the guess and rearrange the sequence of agents in the next iteration. Barman and Verma [8] proved that for a binary XOS function v and its ground set M, a non-wasteful bundle with value no more than $v(M)$ can be computed in time $O(m)$. While the rearrangement process in line 5 is done by sorting n numbers, the time complexity of sorting is known as $O(n \log n)$. The sorting ensures the order of the input of Algorithm 1.

We have completed our proof that Algorithm 2 requires finding non-wasteful bundles for at most mn times and at most m sorting operations. In total, the time complexity of Algorithm 2 is $O(mn(m+\log n))$. □

Theorem 2. *Algorithm 2 returns a $\frac{1}{2}$-APS allocation A in polynomial-time for any instance with binary XOS valuations.*

Proof. In the end of Algorithm 2, each agent a_i gets a bundle with value $\lceil \frac{1}{2}s_i \rceil$. To show that it is a $\frac{1}{2}$-APS allocation, we will prove that throughout this algorithm we have $s_i \ge \mathsf{APS}_i$ for each agent a_i.

Suppose that in some iteration, some agent a_{k+1} cannot obtain a bundle A_{k+1} that satisfies $v_{k+1}(A_{k+1}) = \lceil \frac{1}{2}s_{k+1} \rceil$, we claim that $s_{k+1} > \mathsf{APS}_{k+1}$. By Algorithm 2 and Lemma 1, we have

$$\mathsf{APS}_{k+1} - \left\lfloor b_{k+1} \sum_{i=1}^{k} \left\lceil \frac{1}{2}s_i \right\rceil \right\rfloor < \left\lceil \frac{1}{2}s_{k+1} \right\rceil.$$

Therefore,

$$\begin{aligned}
\mathsf{APS}_{k+1} &< \left\lfloor b_{k+1} \sum_{i=1}^{k} \left\lceil \frac{1}{2}s_i \right\rceil \right\rfloor + \left\lceil \frac{1}{2}s_{k+1} \right\rceil \\
&\le \left\lfloor b_{k+1} \sum_{i=1}^{k} \frac{\lceil \frac{1}{2}s_i \rceil}{b_i} b_i \right\rfloor + \left\lceil \frac{1}{2}s_{k+1} \right\rceil \\
&\le \left\lfloor b_{k+1} \sum_{i=1}^{k} \frac{\lceil \frac{1}{2}s_{k+1} \rceil}{b_{k+1}} b_i \right\rfloor + \left\lceil \frac{1}{2}s_{k+1} \right\rceil \qquad (4) \\
&\le \left\lfloor \sum_{i=1}^{k} b_i \left\lceil \frac{1}{2}s_{k+1} \right\rceil \right\rfloor + \left\lceil \frac{1}{2}s_{k+1} \right\rceil \\
&\le \left\lceil \frac{1}{2}s_{k+1} \right\rceil - 1 + \left\lceil \frac{1}{2}s_{k+1} \right\rceil \qquad (5) \\
&\le s_{k+1}.
\end{aligned}$$

Inequality 4 holds since $\frac{\lceil \frac{1}{2}s_i \rceil}{b_i} \le \frac{\lceil \frac{1}{2}s_{k+1} \rceil}{b_{k+1}}$ when $i < k+1$, and inequality 5 holds since $k < n$ so that $\sum_{i=1}^{k} b_i < 1$.

Therefore, when some agent a_i could not collect a $\frac{1}{2}$-APS bundle from the remaining goods, it holds that $s_i > \mathsf{APS}_i$. This indicates that the decrease of s_i in line 8 would keep $s_i \ge \mathsf{APS}_i$ for every a_i. So in the end of Algorithm 2, $v_i(A_i) = \lceil \frac{1}{2}s_i \rceil \ge \lceil \frac{1}{2}\mathsf{APS}_i \rceil$, which shows that Algorithm 2 is $\frac{1}{2}$-APS. □

Theorem 2 implies the existence of $\frac{1}{2}$-MMS. Ghodsi et al. [17] provided an instance with binary XOS valuations to show that the upper bound of MMS approximation ratio is $\frac{1}{2}$. We improve the lower bound to $\frac{1}{2}$, which makes the bounds match.

Corollary 1. *There exists a $\frac{1}{2}$-MMS allocation under binary XOS valuations, which can be computed in polynomial time.*

Proof. Babaioff et al. [6] proved that when all agents have equal entitlements $\frac{1}{n}$, the APS value is no less than MMS value with any non-negative valuations.

Our Theorem 2 holds for arbitrary entitlement, so when the agents' entitlements are equal, the existence of a $\frac{1}{2}$-APS allocation also holds and we still have a polynomial-time algorithm to compute it. This implies the existence of $\frac{1}{2}$-MMS allocation and a polynomial-time algorithm. □

4 WMMS WITH GENERAL XOS VALUATIONS

In this section, we first show that APS is no longer an upper bound of WMMS, if agents have asymmetric entitlements. In fact, there are instances where WMMS is arbitrarily larger than APS, and thus the algorithms in the previous section do not have any approximation guarantee on WMMS.

Proposition 1. *For any $\delta > 0$, there exists a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ with additive valuations where there is an agent $a_i \in N$ that*

$$\frac{\mathsf{APS}_i}{\mathsf{WMMS}_i} < \delta.$$

Proof. We prove by constructing a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ as follows. There are n agents and $m = n$ items to be allocated. Each agent $a_i \in N$ has the same additive valuation

$$v_i(\{g_j\}) = \begin{cases} \varepsilon, & j \in [n-1], \\ 1-(n-1)\varepsilon, & j = n, \end{cases}$$

where $\varepsilon > 0$ is a small value to be determined. The entitlements of the agents are

$$b_1 = b_2 = \cdots = b_{n-1} = \varepsilon, b_n = 1-(n-1)\varepsilon.$$

It is obvious that the WMMS partition of all the agents is to give each of the first $n-1$ agents one item in $\{g_1, g_2, \ldots, g_{n-1}\}$ and to give a_n the item g_n. Therefore, the WMMS value of a_n is

$$\mathsf{WMMS}_n = 1-(n-1)\varepsilon.$$

However, as to the APS value, consider the price

$$p_1 = p_2 = \cdots = p_{n-1} = 0, p_n = 1.$$

Now a_n could not afford the item g_n, so

$$\mathsf{APS}_n \le (n-1)\varepsilon.$$

For any $\delta > 0$, take

$$\varepsilon < \frac{1}{n-1}\frac{\delta}{\delta+1},$$

then for agent a_n,

$$\frac{\mathsf{APS}_n}{\mathsf{WMMS}_n} \le \frac{(n-1)\varepsilon}{1-(n-1)\varepsilon} < \delta,$$

which proves the proposition. □

Farhadi et al. [10] showed the existence of $\frac{1}{n}$-WMMS allocation for agents with general additive valuations and an instance where no allocation better than $\frac{1}{n}$-WMMS approximation exists. In the following, we show that a $\frac{1}{n}$-WMMS allocation exists for agents with general XOS valuations, which extends the previous result. As to the upper bound, we prove that no algorithm can be better than $\frac{1}{n}$-WMMS for binary XOS valuations. However, if all agents have binary additive valuations, an exact WMMS allocation can be found in polynomial time.

4.1 Computing $\frac{1}{n}$-WMMS Allocations

For a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$, without loss of generality, let $b_1 \ge b_2 \ge \cdots \ge b_n$. Denote by S^i the WMMS partition of a_i as in Definition 3. It is easy to find that

$$v_i(S^i_j) \ge \mathsf{WMMS}_i, \forall j \in [i], i \in [n].$$

Therefore, among the WMMS partition of a_i, there are at least i bundles which satisfy a_i. This observation leads to Algorithm 3. Intuitively, in Algorithm 3, we adopt a Round-Robin-like process, where every agent obtains an item in a round. Then in the first round, before a_i's turn, only $(i-1)$ items have been taken. So each agent a_i can choose a complete bundle in $\{S^i_j, j \in [i]\}$ as her target in the first round. Overall, each agent has a bundle that satisfies her WMMS and she is the first one to select items in the bundle during the Round-Robin-like process. With the help of the additive function that maximizes the XOS valuation for the bundle, each agent can get a bundle of at least $\frac{1}{n}$ of her WMMS.

Inheriting the notation of Definition 3, now we give a concrete description of Algorithm 3.

Algorithm 3 $\frac{1}{n}$-WMMS Existence for General XOS Valuations

```
Require: An instance with general XOS valuations $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$, the WMMS value $(\mathsf{WMMS}_i)_{i\in[n]}$ and the WMMS partition $(S^i)_{i\in[n]}$.
Ensure: A $\frac{1}{n}$-WMMS allocation $A = (A_1, A_2, \ldots, A_n)$.
Rename the agents so that $b_1 \ge b_2 \ge \cdots \ge b_n$.
Initialize $n$ empty bundles $(A_i \leftarrow \emptyset)_{i\in[n]}$.
Set $r \leftarrow 1$.                                   ▷ Round counter.
while $N \ne \emptyset$ do
    for $i \leftarrow 1, 2, \ldots, n$ do
        if $r = 1$ then
            Select a bundle $B^i$ from $(S^i_j)_{j\in[i]}$, where $g_1, g_2, \ldots, g_{i-1} \notin B^i$.   ▷ The bundle is still complete.
            Suppose $v_i(S) = \max_{t\in L} l_t(S)$ for $S \subseteq M$, where $(l_t)_{t\in L}$ are additive functions. Let $t_0 \leftarrow \arg\max_{t\in L} l_t(B^i)$ and $l^i \leftarrow l_{t_0}$.
            Set $g_i \leftarrow \arg\max_{e\in B^i} l^i(e)$.
        if $a_i \in N$ then
            if $B^i \ne \emptyset$ then              ▷ Round-Robin-like process
                Set $g \leftarrow \arg\max_{e\in B^i} l^i(e)$.
                Set $A_i \leftarrow A_i \cup \{g\}$.
                Set $B^j \leftarrow B^j \setminus \{g\}$ for $j \in [n]$, if $B^j$ is already selected.
                Set $M \leftarrow M \setminus \{g\}$.
            else
                Set $N \leftarrow N \setminus \{a_i\}$.       ▷ Agent $a_i$ finishes.
    Set $r \leftarrow r + 1$.
if $M \ne \emptyset$ then
    Allocate the remaining goods arbitrarily.
return $A = (A_1, A_2, \ldots, A_n)$.
```

Theorem 3. *For any instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ with XOS valuations, there exists a $\frac{1}{n}$-WMMS allocation.*

Proof. Algorithm 3 admits the existence of $\frac{1}{n}$-WMMS allocation. First we claim that

$$v_i(S^i_j) \ge \mathsf{WMMS}_i, \forall j \in [i], i \in [n].$$

Recall the definition 3 of WMMS partition, the WMMS partition of a_i is

$$S^i = \arg\max_{S\in\Pi_n} \min_{j\in[n]} v_i(S^i_j)\frac{b_i}{b_j},$$

and S^i_j is the bundle a_i wants to assign to a_j. By the definition 2 of WMMS,

$$v_i(S^i_j)\frac{b_i}{b_j} \ge \mathsf{WMMS}_i, \forall j \in [n].$$

With the assumption that $b_1 \geq b_2 \geq \cdots \geq b_n$, we have $v_i(S^i_j) \geq \mathsf{WMMS}_i$ for $j \in [i]$.

This means that every agent a_i starts from a bundle B^i where $v_i(B^i) \geq \mathsf{WMMS}_i$ in line 7. As each agent could only get one item in a round, before the first round of a_i, only $(i-1)$ items have been taken away by the previous agents. Therefore, the i bundles $(S^i_j)_{j\in[i]}$ for a_i could guarantee the existence of B^i.

As to the approximation ratio, the agent a_i is the first one to take goods from the bundle B^i, and she will always select the most valuable one according to the additive function l^i as in line 8. Let $l^i|_{B^i}$ be an additive function where

$$l^i|_{B^i}(e) = \begin{cases} l^i(e), & e \in B^i, \\ 0, & e \notin B^i. \end{cases}$$

Suppose a_i receives some item g^r at the r-th round, then by the selection of line 12, we have $l^i|_{B^i}(g^r) \geq l^i|_{B^i}(e^r)$ for all items e^r that any other agent could get between the r-th round and the $(r+1)$-th round of a_i. As no agent takes any item from B^i until a_i does in the first round and there are n agents in total, this process indicates

$$l^i(A_i) = l^i|_{B^i}(A_i) \geq \frac{1}{n} l^i|_{B^i}(B^i) = \frac{1}{n} l^i(B^i).$$

By the definition of XOS functions, $v_i(S) \geq l^i(S), \forall S \subseteq M$. In summary, we have

$$v_i(A_i) \geq l^i(A_i) \geq \frac{1}{n} l^i(B^i) = \frac{1}{n} v_i(B^i) \geq \frac{1}{n} \mathsf{WMMS}_i.$$

Therefore, the allocation A is a $\frac{1}{n}$-WMMS allocation for the instance. □

4.2 Upper Bound

The tight example exploits the fact that the agent with the largest entitlement may only have one bundle satisfying her WMMS.

Theorem 4. *For any $n \geq 2$, there exists a fair allocation instance of n agents with binary XOS valuations where no allocation is better than $\frac{1}{n}$-WMMS.*

Proof. Let n be the number of agents, we construct a fair allocation instance that no more than $\frac{1}{n}$-WMMS allocation exists. Consider an instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ as follows. Here we set $m = 2n-1$ and $M = \{g_1, g_2, \ldots, g_{2n-1}\}$. For any $S \subseteq M$, the value functions of agents are

$$\begin{aligned} v_1(S) =& v_2(S) = \cdots = v_{n-1}(S) \\ =& \max\{|S \cap \{g_1\}|, |S \cap \{g_2\}|, \ldots, |S \cap \{g_n\}|\}, \\ v_n(S) =& \max\{|S \cap \{g_1, g_2, \ldots, g_n\}|, \\ & |S \cap \{g_{n+1}\}|, |S \cap \{g_{n+2}\}|, \ldots, |S \cap \{g_{2n-1}\}|\}. \end{aligned}$$

The entitlements of the agents are

$$b_1 = b_2 = \cdots = b_{n-1} = \frac{1}{2n-1}, b_n = \frac{n}{2n-1}.$$

It is straightforward to observe that the valuations are binary XOS. We then calculate the WMMS value of the n agents. In spite of the imbalance in the entitlement, the only reasonable allocation for $(a_i)_{i\in[n-1]}$ is to give each of the n agents one item in $\{g_1, g_2, \ldots, g_n\}$ and allocate the remaining items arbitrarily. Otherwise, there will be at least one agent receiving a zero-valued bundle, which makes her minimum value in WMMS definition be 0. By definition 2,

$$\mathsf{WMMS}_1 = \mathsf{WMMS}_2 = \cdots = \mathsf{WMMS}_{n-1} = \frac{1}{n}.$$

For a_n, since she has n times as much entitlement as the others, according to her value function, her favorable allocation is to receive $\{g_1, g_2, \ldots, g_n\}$ herself and give every other agent an item in $\{g_{n+1}, g_{n+2}, \ldots, g_{2n-1}\}$. Therefore, $\mathsf{WMMS}_n = n$.

As to the allocation, to ensure that the value every agent receives is greater than 0, each of the first $n-1$ agents must obtain at least an item in $\{g_1, g_2, \ldots, g_n\}$. Without loss of generality, we will assign the items $\{g_1, g_2, \ldots, g_{n-1}\}$ to the first $n-1$ agents, which makes agent n could only receive a bundle $S \subseteq \{g_n, g_{n+1}, \ldots, g_{2n-1}\}$ with value 1. In summary, either for some agent in $(a_i)_{i\in[n-1]}$, the allocation is 0-WMMS, or for a_n the allocation is at most $\frac{1}{n}$-WMMS, which finishes the proof. □

4.3 Binary Additive Valuations

In contrast to the $\frac{1}{n}$-WMMS bound for general additive valuations [10] and binary XOS valuations, we present a polynomial-time algorithm which outputs an exact WMMS allocation for agents with binary additive valuations. For a fair allocation instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ with binary additive valuations, let

$$D^i = \{g \in M | v_i(g) = 1\},$$

then the valuation of agent a_i is $v_i(S) = |S \cap D^i|, \forall S \subseteq M$.

We begin with Algorithm 4, which computes the WMMS partition, hence the WMMS value, of any agent in polynomial time.

Algorithm 4 WMMS Partition for Binary Additive Valuations

```
Require: An instance with binary additive valuations
         $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ and an agent $a_i \in N$.
Ensure: A WMMS partition $S^i = (S^i_j)_{j\in[n]}$ of agent $a_i$.
Initialize $n$ empty bundles $(S^i_j \leftarrow \varnothing)_{j\in[n]}$.
Let $D^i \leftarrow \{g \in M | v_i(g) = 1\}$.
for $g \in D^i$ do
    Set $j_{\min} \leftarrow \arg\min_{j\in[n]} \frac{|S^i_j|}{b_j}$, break tie arbitrarily.
    Set $S^i_{j_{\min}} \leftarrow S^i_{j_{\min}} \cup \{g\}$.
Allocate goods in $M \setminus D_i$ arbitrarily.
return $S^i = (S^i_1, S^i_2, \ldots, S^i_n)$.
```

Lemma 3. *For any instance $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ with binary additive valuations, Algorithm 4 returns a WMMS partition of agent a_i in polynomial time.*

Proof. After the r-th step of the for loop, denote the n bundles we assigned are $S^{i,r} = (S^{i,r}_j)_{j\in[n]}$, then the set of items we have assigned are $M^{i,r} = \cup_{j\in[n]} S^{i,r}_j$.

Let $L^{i,r}$ denote the bundle whose value to agent a_i, divided by the entitlement b_j corresponding to its index j, is the smallest, i.e.,

$$L^{i,r} = \min_{j\in[n]} \frac{|S^{i,r}_j|}{b_j}.$$

We claim that for all the for loop index r, $L^{i,r}$ satisfies

$$\frac{|S_j^{i,r}| - 1}{b_j} \leq L^{i,r} \leq \frac{|S_j^{i,r}|}{b_j}, \forall j \in [n]. \tag{6}$$

The right side of inequality 6 definitely holds according to the definition of $L^{i,r}$. We will prove the left side of inequality 6 by contradiction. We assume that there is a bundle $S_k^{i,r}$ satisfies $\frac{|S_k^{i,r}|-1}{b_k} > L^{i,r}$ and that the last item in bundle $S_k^{i,r}$ was assigned to it during the r'-th for loop. After r'-th round, there is

$$\frac{|S_k^{i,r'}|}{b_k} = \frac{|S_k^{i,r}| - 1}{b_k} > L^{i,r} \geq L^{i,r'}.$$

The last inequality holds since, according to Algorithm 4, the number of items in each bundle is non-decreasing, and then the value of $L^{i,r}$ is non-decreasing in r. Since $\frac{|S_k^{i,r'}|}{b_k}$ is strictly larger than $L^{i,r'}$, in this round, the item will not assigned to $S_k^{i,r'}$ according to Algorithm 4, which contradicts our assumption. We have proved the left side of Inequality 6.

Next, we will show that for any for loop index $r \in [|D^i|]$, the set of bundles $S^{i,r}$ is a WMMS partition for the set of assigned items $M^{i,r}$ for agent a_i. Therefore, by the definition of WMMS, we have

$$\mathsf{WMMS}_i(M^{i,r}) = L^{i,r} b_i.$$

We will also prove this by contradiction. We assume that partition $S^{i,r}$ is not a WMMS partition for item set $M^{i,r}$. Let a WMMS partition be $T^{i,r} = (T_j^{i,r})_{j\in[n]}$. Therefore, by the definition of WMMS partition, we have $\min_{j\in[n]} \frac{|T_j^{i,r}|}{b_j} > L^{i,r}$. Since $T^{i,r}$ is not equal to $S^{i,r}$, there must be a bundle index k such that $|T_k^{i,r}| \leq |S_k^{i,r}| - 1$. According to Inequality 6, it means

$$\frac{|T_k^{i,r}|}{b_k} \leq \frac{|S_k^{i,r}| - 1}{b_k} \leq L^{i,r},$$

which leads a contradiction.

This finishes the proof for the correctness of Algorithm 4.

For the time complexity of Algorithm 4, we need to iterate through all the m items, and each iteration costs a binary search in n values. Overall, the time complexity is $O(m \log n)$. □

It is straightforward to observe that the calculation of the WMMS partition serves as the ”worst” case allocation for each agent. That is, if we let each agent select items in her own favor according to the rule in Algorithm 4, then from the point of view of a certain agent, there will be a possibility that she is indifferent to some items taken by the other agents. This observation leads to Algorithm 5.

Theorem 5. *Algorithm 5 returns a WMMS allocation in polynomial time for any fair allocation instance* $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$ *with binary additive valuations.*

Proof. For any agent a_i, we have $v_i(A_i) \geq \mathsf{WMMS}_i$. That is because if everybody takes the item g from the set D_i in each while loop, Algorithm 5 does the same work as Algorithm 4. In this case, Lemma 3 have proved that $v_i(A_i) \geq \mathsf{WMMS}_i$. Otherwise, if some agent gets an item not in D_i, the remaining items in D_i will be increased, and the number of items will be increased at the same time. We still have $|A_i| = v_i(A_i) \geq \mathsf{WMMS}_i$. The time complexity of Algorithm 5 is also $O(m \log n)$. □

Algorithm 5 WMMS Allocation for Binary Additive Valuations

```
Require: An instance with binary additive valuations
         $(N, M, (v_i)_{i\in[n]}, (b_i)_{i\in[n]})$.
Ensure: A WMMS partition $A = (A_1, A_2, \dots, A_n)$.
Initialize $n$ empty bundles $(A_i \leftarrow \emptyset)_{i\in[n]}$.
Let $D^i \leftarrow \{g \in M | v_i(g) = 1\}, \forall i \in [n]$.
while $N \neq \emptyset$ do
    Set $i_{\min} \leftarrow \arg\min_{i\in[n]} \frac{|A_i|}{b_i}$.
    if $D^{i_{\min}} \neq \emptyset$ then
        Select an arbitrary item $g \in D^{i_{\min}}$ and set $A_i \leftarrow A_i \cup \{g\}$.
        Set $D^i \leftarrow D^i \setminus \{g\}, \forall i \in [n]$.
        Set $M \leftarrow M \setminus \{g\}$.
    else
        $N \leftarrow N \setminus \{a_{i_{\min}}\}$.
Allocate remaining goods in $M$ arbitrarily.
return $S^i = (S_1^i, S_2^i, \dots, S_n^i)$.
```

5 CONCLUSION REMARKS

In this paper, we prove the tight approximation guarantees for asymmetric fair allocation with binary XOS valuations under APS and WMMS fairness criteria. Specifically, we design a polynomial-time algorithm that computes a $\frac{1}{2}$-approximate APS allocation, which matches the upper bound of the approximation ratio. This result also implies the existence of a $\frac{1}{2}$-approximate MMS allocation when agents have equal entitlements. When agents have different entitlements, we show that APS can be arbitrarily smaller than WMMS, even when agents have additive valuations. We then design an algorithm that ensures a $\frac{1}{n}$-WMMS allocation and prove that a better than $\frac{1}{n}$ approximation ratio is not guaranteed. The approximation ratio holds even for general XOS valuations. In addition, we show that an exact WMMS allocation can be achieved for binary additive valuations; this result serves to highlight that obtaining a $\frac{1}{n}$-WMMS guarantee under binary XOS valuations is non-trivial. There are several interesting future research directions. First, it is still unknown if constant approximations can be ensured if agents have subadditive valuations with binary marginals. Second, for APS/WMMS with asymmetric agents under binary submodular (matroid-rank) valuations, while our binary XOS results already apply, it remains open whether strictly better approximation guarantees can be achieved beyond the symmetric-agent setting. Third, in the current work, we have solely focused on the allocation of goods. It is interesting to investigate the minor problem of chores.

ACKNOWLEDGMENTS

We thank Yiwei Gao and Ankang Sun for helpful discussions. Ziheng Chen, Zihan Luo, and Jialin Zhang are supported in part by the National Natural Science Foundation of China (Grant No. 62272441). Bo Li is supported in part by the Hong Kong SAR Research Grants Council (Grant No. PolyU 15224823) and the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2024A1515011524).

REFERENCES

[1] Hannaneh Akrami and Jugal Garg. 2024. Breaking the 3/4 barrier for approximate maximin share. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 74–91.

[2] Hannaneh Akrami, Jugal Garg, Eklavya Sharma, and Setareh Taki. 2023. Simplification and improvement of MMS approximation. *arXiv preprint arXiv:2303.16788* (2023).

[3] Hannaneh Akrami, Kurt Mehlhorn, Masoud Seddighin, and Golnoosh Shahkarami. 2023. Randomized and deterministic maximin-share approximations for fractionally subadditive valuations. *Advances in Neural Information Processing Systems* 36 (2023), 58821–58832.

[4] Haris Aziz, Hau Chan, and Bo Li. 2019. Maxmin share fair allocation of indivisible chores to asymmetric agents. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 1787–1789.

[5] Haris Aziz, Gerhard Rauchecker, Guido Schryen, and Toby Walsh. 2017. Algorithms for max-min share fair allocation of indivisible chores. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[6] Moshe Babaioff, Tomer Ezra, and Uriel Feige. 2024. Fair-share allocations for agents with arbitrary entitlements. *Mathematics of Operations Research* 49, 4 (2024), 2180–2211.

[7] Siddharth Barman and Sanath Kumar Krishnamurthy. 2020. Approximation algorithms for maximin fair division. *ACM Transactions on Economics and Computation* 8, 1 (2020), 1–28.

[8] Siddharth Barman and Paritosh Verma. 2021. Approximating Nash social welfare under binary XOS and binary subadditive valuations. In *International Conference on Web and Internet Economics*. Springer, 373–390.

[9] Eric Budish. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy* 119, 6 (2011), 1061–1103.

[10] Alireza Farhadi, Mohammad Ghodsi, Mohammad Taghi Hajiaghayi, Sebastien Lahaie, David Pennock, Masoud Seddighin, Saeed Seddighin, and Hadi Yami. 2019. Fair allocation of indivisible goods to asymmetric agents. *Journal of Artificial Intelligence Research* 64 (2019), 1–20.

[11] Uriel Feige. 2025. From multi-allocations to allocations, with subadditive valuations. *arXiv preprint arXiv:2506.21493* (2025).

[12] Uriel Feige and Vadim Grinberg. 2025. Fair allocations with subadditive and XOS valuations. In *Proceedings of the 26th ACM Conference on Economics and Computation*. 160–185.

[13] Uriel Feige and Shengyu Huang. 2025. Concentration and maximin fair allocations for subadditive valuations. *arXiv preprint arXiv:2502.13541* (2025).

[14] Uriel Feige and Xin Huang. 2023. On picking sequences for chores. In *Proceedings of the 24th ACM Conference on Economics and Computation*. 626–655.

[15] Uriel Feige, Ariel Sapir, and Laliv Tauber. 2021. A tight negative example for MMS fair allocations. In *International Conference on Web and Internet Economics*. Springer, 355–372.

[16] Jugal Garg and Setareh Taki. 2020. An improved approximation algorithm for maximin shares. In *Proceedings of the 21st ACM Conference on Economics and Computation*. 379–380.

[17] Mohammad Ghodsi, MohammadTaghi HajiAghayi, Masoud Seddighin, Saeed Seddighin, and Hadi Yami. 2018. Fair allocation of indivisible goods: Improvements and generalizations. In *Proceedings of the 19th ACM Conference on Economics and Computation*. 539–556.

[18] Ehsan Heidari, Alireza Kaviani, Masoud Seddighin, and AmirMohammad Shahrezaei. 2025. Improved maximin share guarantee for additive valuations. *arXiv preprint arXiv:2510.10423* (2025).

[19] Xin Huang and Pinyan Lu. 2021. An Algorithmic Framework for Approximating Maximin Share Allocation of Chores. In *Proceedings of the 22nd ACM Conference on Economics and Computation*. ACM, 630–631.

[20] Xin Huang and Erel Segal-Halevi. 2023. A Reduction from Chores Allocation to Job Scheduling. In *Proceedings of the 24th ACM Conference on Economics and Computation*. ACM, 908.

[21] Halvard Hummel. 2025. Maximin shares in hereditary set systems. *ACM Transactions on Economics and Computation* 13, 3 (2025), 1–33.

[22] Pooja Kulkarni, Rucha Kulkarni, and Ruta Mehta. 2024. Approximating APS Under Submodular and XOS Valuations with Binary Marginals. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, Mehdi Dastani, Jaime Simão Sichman, Natasha Alechina, and Virginia Dignum (Eds.). International Foundation for Autonomous Agents and Multiagent Systems / ACM, 1057–1065.

[23] David Kurokawa, Ariel D Procaccia, and Junxing Wang. 2018. Fair enough: Guaranteeing approximate maximin shares. *J. ACM* 65, 2 (2018), 1–27.

[24] Bo Li, Yingkai Li, and Xiaowei Wu. 2022. Almost (Weighted) Proportional Allocations for Indivisible Chores. In *Proceedings of the ACM Web Conference*. 122–131.

[25] Bo Li, Fangxiao Wang, and Yu Zhou. 2023. Fair Allocation of Indivisible Chores: Beyond Additive Costs. In *Conference and Workshop on Neural Information Processing Systems*.

[26] Zhentao Li and Adrian Vetta. 2021. The fair division of hereditary set systems. *ACM Transactions on Economics and Computation* 9, 2 (2021), 1–19.

[27] Ariel D Procaccia and Junxing Wang. 2014. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the 15th ACM conference on Economics and computation*. 675–692.

[28] Masoud Seddighin and Saeed Seddighin. 2024. Improved maximin guarantees for subadditive and fractionally subadditive fair allocation problem. *Artificial Intelligence* 327 (2024), 104049.

[29] Masoud Seddighin and Saeed Seddighin. 2025. Beating the Logarithmic Barrier for the Subadditive Maximin Share Problem. In *Proceedings of the 26th ACM Conference on Economics and Computation*. 764–782.

[30] Hugo Steinhaus. 1949. Sur la division pragmatique. *Econometrica: Journal of the Econometric Society* (1949), 315–319.

[31] Gilad Ben Uziahu and Uriel Feige. 2023. On fair allocation of indivisible goods to submodular agents. *arXiv preprint arXiv:2303.12444* (2023).

[32] Vignesh Viswanathan and Yair Zick. 2023. A general framework for fair allocation under matroid rank valuations. In *Proceedings of the 24th ACM Conference on Economics and Computation*. 1129–1152.

[33] Fangxiao Wang, Bo Li, and Pinyan Lu. 2024. Improved approximation of weighted MMS fairness for indivisible chores. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence*. 3014–3022.