Constant Approximation for Weighted Nash Social Welfare with Submodular Valuations

Yuda Feng [*1], Yang Hu [2], Shi Li [*1], and Ruilong Zhang [3]

[1]School of Computer Science, Nanjing University, Nanjing, China
[2]Institute for Interdisciplinary Information Sciences, Tsinghua University, Peking, China
[3]Department of Mathematics, Technical University of Munich, Munich, Germany
`yudafeng@smail.nju.edu.cn`, `y-hu22@mails.tsinghua.edu.cn`, `shili@nju.edu.cn`,
`ruilong.zhang@tum.de`

Abstract

We study the problem of assigning items to agents so as to maximize the *weighted* Nash Social Welfare (NSW) under submodular valuations. The best-known result for the problem is an $O(nw_{\max})$-approximation due to Garg, Husic, Li, Vega, and Vondrak [13], where $w_{\max}$ is the maximum weight over all agents. Obtaining a constant approximation algorithm is an open problem in the field that has recently attracted considerable attention.

We give the first such algorithm for the problem, thus solving the open problem in the affirmative. Our algorithm is based on the natural Configuration LP for the problem, which was introduced recently by Feng and Li [11] for the additive valuation case. Our rounding algorithm is similar to that of Li [25] developed for the unrelated machine scheduling problem to minimize weighted completion time. Roughly speaking, we designate the largest item in each configuration as a large item and the remaining items as small items. So, every agent gets precisely 1 fractional large item in the configuration LP solution. With the rounding algorithm in [25], we can ensure that in the obtained solution, every agent gets precisely 1 large item, and the assignments of small items are negatively correlated.

1 Introduction

We study the problem of allocating a set M of indivisible items among a set N of agents, where each agent $i \in N$ has a monotone non-negative submodular valuation $v_i : 2^M \to \mathbb{R}_{\geq 0}$ and a weight $w_i \in (0, 1)$ with $\sum_{i \in N} w_i = 1$. The *weighted* Nash Social Welfare (NSW) problem under submodular valuations asks for partition $\mathcal{S} := (S_i)_{i \in N}$ of M that maximizes the weighted geometric mean of the agents' valuations:

$$\mathsf{NSW}(\mathcal{S}) = \prod_{i \in N} (v_i(S_i))^{w_i}.$$

The case when all w_i's are equal to $1/n$ is called the unweighted Nash Social Welfare problem. As usual, we assume we are given a value oracle for each v_i. W.l.o.g, we assume $v_i(\emptyset) = 0$ for every agent $i \in N$ [1].

Fair and efficient allocation of resources is a central problem in computer science, game theory, and social choices, with applications across diverse domains [1, 3, 4, 22, 27, 31, 32, 37]. Three distinct communities independently discovered the notation of Nash social welfare: as a solution to the bargaining problem in classical game theory [28], as a well-established concept of proportional fairness in networking [23], and as the celebrated notion of competitive equilibrium with equal incomes in economics [35]. The unweighted case for the problem was introduced by Nash [28], and it was later extended to the weighted case [17, 21]. This extension has since been widely studied and applied across various fields, such as bargaining theory [7, 24, 34], water allocation [9, 19], climate agreements [38], and more. One of the most important features of the NSW objective is that it offers a tradeoff between the frequently conflicting demands of fairness and efficiency.

A special case for the valuations v_i is when they are additive. The unweighted NSW problem with additive valuations is an important topic in optimization and has received considerable interest. Barman, Krishnamurthy, and Vaish [2] developed a ($\mathsf{e}^{1/\mathsf{e}} \approx 1.445$)-approximation algorithm that finds an allocation that is both Pareto-efficient and envy-free up to one item (EF1). They showed that this problem can be reduced to the case of identical valuations, where any EF1 allocation can achieve an approximation ratio of $\mathsf{e}^{1/\mathsf{e}} \approx 1.445$. On the negative side, Garg, Hoefer, and Mehlhorn [12] established a hardness of $\sqrt{8/7}$.

For the weighted case with additive valuations, Brown, Laddha, Pittu, and Singh [5] introduced an approximation algorithm with a ratio of $5 \cdot \exp(2 \log n + 2 \sum_{i \in A} w_i \log w_i)$. Later, Feng and Li [11] presented an elegant ($\mathsf{e}^{1/\mathsf{e}} + \epsilon$)-approximation algorithm for the weighted case, using their novel configuration LP and the Shmoys-Tardos rounding procedure developed in the context of unrelated machine scheduling. The approximation ratio matches the best-known ratio for the unweighted case.

When the n valuation functions are additive and identical, Nguyen and Rothe [29] developed a PTAS for the unweighted NSW problem. Later, Inoue and Kobayashi [20] gave an additive PTAS for the problem, i.e., a polynomial-time algorithm that maximizes the Nash social welfare within an additive error of $\epsilon v_{\max}$, where $v_{\max}$ is the maximum utility of an item.

Li and Vondrak [26] developed the first constant approximation algorithm for unweighted NSW with submodular valuations using convex programming. The ratio has been improved by Garg, Husic, Li, Vega, and Vondrak [13] to $(4 + \epsilon)$ using an elegant local-search-based algorithm. When additionally $n = O(1)$, by guessing the value and the $O(1)$ largest items for each agent, and using the multilinear extension of submodular functions, a $\mathsf{e}/(\mathsf{e} - 1)$-approximation can be achieved [16]. In the same paper, [16] proved that unweighted NSW with submodular valuations is hard to approximate within $\mathsf{e}/(\mathsf{e} - 1) - \epsilon$. The hardness holds even for the case $n = O(1)$.

For the weighted NSW problem with submodular valuations, [13] showed that the approximation ratio of the local search algorithm becomes $O(n w_{\max})$, where $w_{\max} := \max_{i \in [n]} w_i$ is the maximum weight over the agents. In the new version [15] of the paper, the authors presented a $(6\mathsf{e} + \epsilon)$-

---

[1] If $v_i(\emptyset) > 0$ for some $i \in N$, we can create a "private" item j_i for i which has 0-value to all agents other than i. We replace the valuation of i with v_i', which is defined as follows: $v_i'(S) := v_i(S) - v_i(\emptyset)$ if $j_i \notin S$ and $v_i'(S) = v_i(S \setminus j_i)$ if $j_i \in S$.

approximation algorithm with running time $2^{O(n \log n)}\text{poly}(m, 1/\epsilon)$, which is polynomial when $n = O(1)$.

For the more general setting where the valuations are subadditive, Dobzinski, Li, Rubinstein, and Vondrák [10] recently proposed a constant approximation algorithm when agents are unweighted, provided that we have access to demand oracles for the valuation functions.

Our Result. In this paper, we give the first polynomial-time $O(1)$-approximation algorithm for weighted Nash social welfare under the submodular valuations. The best result prior to this work was the $O(nw_{\max})$-approximation due to Garg, Husic, Li, Vega, and Vondrak [13].

Theorem 1.1. *For any $\epsilon > 0$, there is a randomized $(233+\epsilon)$-approximation algorithm for the weighted Nash social welfare problem with submodular valuations, with running time polynomial in the size of the input and $\frac{1}{\epsilon}$.*

For convenience, we list the known approximation results for the NSW problem in Table 1.

	Additive		Submodular		Subadditive	
	LB	UB	LB	UB	LB	UB
Unweighted	$\sqrt{\frac{8}{7}}$ [12]	$\mathsf{e}^{1/\mathsf{e}} + \epsilon$ [2]	$\frac{\mathsf{e}}{\mathsf{e}-1}$ [16]	$4 + \epsilon$ [13]		$O(1)^*$ [10]
Weighted		$\mathsf{e}^{1/\mathsf{e}} + \epsilon$ [11]		$233 + \epsilon$ (Theorem 1.1)		

Table 1: Known Results for Nash social welfare. LB and UB stand for lower and upper bounds, respectively. The result with * requires demand oracles for valuation functions. When the function is identical additive, the upper bound for the unweighted case is PTAS [20, 29]. When $n = O(1)$, the upper bounds for unweighted and weighted NSW with submodular valuations are respectively $\mathsf{e}/(\mathsf{e} - 1)$ [16] and $6\mathsf{e} + \epsilon$ [15].

1.1 Overview of Our Techniques

Our algorithm leverages the configuration LP introduced in [11] for the additive valuation case. For each agent $i \in N$ and subset of items $S \subseteq M$, we define a variable $y_{i,S}$ to indicate whether the set of items assigned to i is precisely S. The objective of this LP is to minimize $\sum_{i,S} y_{i,S} \cdot w_i \cdot \ln v_i(S)$, the logarithm of the NSW objective. After solving the LP, we apply the rounding procedure from [25], developed for the weighted completion time minimization problem in the unrelated machine scheduling setting. Then we prove concentration bounds for the values obtained by each agent, using arguments developed for pipage rounding.

To build intuition, let us focus on the unweighted case. For the special case where $|M| = |N|$, the problem reduces to a maximum-weight bipartite matching problem with weights given by the logarithm of values. So, any general algorithm for the problem must capture the maximum weight of the bipartite matching algorithm as a special case. Interestingly, previous results showed that if one is given the largest (i.e., the most valuable) item assigned to every agent, then an $O(1)$-approximation algorithm is easy to obtain using local search [13] or LP rounding [14, 26]. For example, with this idea, Garg, Husic, Li, Vega, and Vondrak [13] designed an elegant 4-approximation local search algorithm. They first compute an initial matching of one item to every agent so as to maximize the NSW objective, then assign the remaining items using local search with an endowed valuation function, and finally rematch the initially assigned items to agents to maximize the final Nash social welfare. Unfortunately, their algorithm fails to give an $O(1)$-approximation when the agents are weighted.

Our algorithm implements the idea of "matching largest items to agents" using the configuration LP solution as a guide. We achieve a per-client guarantee, allowing us to give an $O(1)$-approximation for the *weighted* NSW problem with submodular valuations. After obtaining an LP solution $(y_{i,S}^*)_{i,S}$,

for each agent i and configuration S, we designate the largest item in S as a "large" item for i, while treating the remaining items as "small". This creates a fractional assignment in which each agent receives exactly one fractional large item. While maintaining marginal probabilities in our rounding algorithm, we ensure that each agent gets exactly one large item, and the assignment of small items are negatively correlated. That is, we select a random matching for large items.

If the large and small items were disjoint, the rounding algorithm would be straightforward. However, complications arise when an item may be large for one agent and small for another — or even for the same agent in different configurations. This necessitates a correlated assignment strategy for large and small items. This is where we employ the iterative rounding procedure of [25]. We construct a bipartite multi-graph between agents and items, with two edge types: *marked* edges for large items and *unmarked* edges for small items. During iterative rounding, we identify either a simple cycle of marked edges or a pseudo-marked path – a simple path of marked edges with two unmarked edges at the ends – and apply rotation or shifting operations on the cycle or path in each iteration. This process ultimately yields an integral assignment.

To analyze the approximation ratio, we focus on each agent i and analyze $\mathbb{E}[\ln(v_i(T))]$, where T is the set of items assigned to i. Note that T includes exactly one large item, respecting the marginal probabilities. Let T^{S} denote the remaining items, i.e., the small items assigned to i. The assignments of the large item and the small items may be positively correlated, so we analyze the worst-case scenario for this correlation. However, the assignments of the small items are negatively correlated; more precisely, they are determined through a pipage-rounding procedure. Using the concave pessimistic estimator technique from [18] and the submodularity of the function v_i, we can establish concentration bounds for $v_i(T^{\mathrm{S}})$. With the bounds, we can lower bound $\mathbb{E}[\ln(v_i(T))]$ by $\sum_S y^*_{i,S} \ln v_i(S) - O(1)$.

Organization. The rest of the paper is organized as follows. We introduce some preliminaries in Section 2, describe our algorithm in Section 3, and give its analysis in Section 4. For a smoother flow in the main text, we defer some proofs to the appendix.

2 Preliminaries

Definition 2.1 (Monotone Submodular Functions)**.** *Let M be the ground set (item set) and $f : 2^M \to \mathbb{R}_{\geq 0}$ be a function defined over M.*

- *f is submodular if for any $S, T \subseteq M$, $f(S) + f(T) \geq f(S \cap T) + f(S \cup T)$.*
- *f is monotone if $f(S) \leq f(T)$ whenever $S \subseteq T$.*

For convenience, we slightly abuse the notation and use $f(j)$ to represent $f(\{j\})$.

2.1 Truncations of Submodular Functions

Definition 2.2 (Truncated Function)**.** *Given a monotone submodular function $f : 2^M \to \mathbb{R}_{\geq 0}$ with $f(\emptyset) = 0$ and a real $R > 0$, define $f^{(R)} : 2^M \to \mathbb{R}_{\geq 0}$ to be the following function:*

$$f^{(R)}(S) := \mathbb{E}_{\theta \sim [0,1]} \left[f \left(\left\{ j \in S : \theta \leq \frac{R}{f(j)} \right\} \right) \right], \quad \forall S \subseteq M.$$

We say $f^{(R)}$ is the function obtained from f by truncating individual values by R.

The truncated function provides a trackable lower bound for the valuation function. After truncating the function, the marginal values will be bounded by R; this allows us to apply the concentration bound. This is given in the following lemma:

Lemma 2.3. *Let $f : 2^M \to \mathbb{R}_{\geq 0}$ be a monotone submodular function and a $R > 0$ be a real. Let $R' \geq R$ be a real and $S \subseteq M$ be a non-empty set such that $\max_{j \in S} f(j) \leq R'$. Then, the following properties are true:*

4

(2.3a) $f^{(R)}$ *is a monotone submodular function such that* $f^{(R)}(j) \leq R$ *for all* $j \in M$.

(2.3b) $f(S) \geq f^{(R)}(S) \geq \frac{R}{R'} \cdot f(S)$.

Proof. It is easy to see that $f^{(R)}$ is a monotone function. The truncated function $f^{(R)}$ can be regarded as a linear combination of submodular functions, and thus, it is also a submodular function. For any $j \in M$, we have $f^{(R)}(j) = \min\{\frac{R}{f(j)}, 1\} \cdot f(j) \leq R$. Thus, *(2.3a)* holds.

For *(2.3b)*, the proof of $f(S) \geq f^{(R)}(S)$ is straightforward. If $\theta \leq \frac{R}{R'}$, then $\{\, j \in S : \theta \leq \frac{R}{f(j)} \,\} = S$ as $f(j) \leq R'$ for all $j \in S$. Thus, $f^{(R)}(S) \geq \frac{R}{R'} \cdot f(S)$. This finishes the proof of Lemma 2.3. □

We remark that when the valuation function is additive, the definition of the truncated function can be simplified: $f^{(R)}(S) := \sum_{j \in S} \min\{R, f(j)\}$.

2.2   Extensions of Submodular Functions

We shall also consider the multilinear and concave (maximum) extensions of a submodular function, which are widely used in the literature. The concave extension is also known as the "Configuration LP" extension in the literature.

Definition 2.4 (Multilinear Extension). *Given a monotone submodular function* $f : 2^M \to \mathbb{R}_{\geq 0}$, *its multilinear extension* $F : [0,1]^M \to \mathbb{R}_{\geq 0}$ *is defined as:*

$$F(\mathbf{x}) = \sum_{S \subseteq M} f(S) \prod_{j \in S} x_j \prod_{j \in (M \setminus S)} (1 - x_j).$$

Definition 2.5 (Concave Extension). *Given a monotone submodular function* $f : 2^M \to \mathbb{R}_{\geq 0}$, *its concave extension* $f^+ : [0,1]^M \to \mathbb{R}_{\geq 0}$ *is:*

$$f^+(\mathbf{x}) = \max \left\{ \sum_{S \subseteq M} \alpha_S f(S) : \sum_{S \subseteq M} \alpha_S \leq 1; \forall S \subseteq M, \alpha_S \geq 0; \forall j \in M, \sum_{S : j \in S} \alpha_S = x_j \right\}.$$

The following relation between the multilinear and concave extensions has been shown in [36, Lemma 3.7, 3.8].

Lemma 2.6 ([36]). *Given a monotone non-negative submodular function* f, *its multilinear extension* F, *and its concave extension* f^+, *and a point* $\mathbf{x} \in [0,1]^M$, *we have* $f^+(\mathbf{x}) \geq F(\mathbf{x}) \geq (1 - \frac{1}{e}) \cdot f^+(\mathbf{x})$.

2.3   Concentration Bound for Submodular Functions in Pipage Rounding

We consider the following modified version of the pipage rounding procedure described in Algorithm 1. This enables us to use the Chernoff-type concentration bounds stated in Theorem 2.7. Suppose we are guaranteed that the algorithm terminates in the finite number of iterations with probability 1.

Algorithm 1 Modified Pipage Rounding

Input: $\mathbf{x}^* \in [0,1]^{\bar{n}}$.

Output: an integral $\mathbf{x} \in \{0,1\}^{\bar{n}}$.

1: $\mathbf{x} \leftarrow \mathbf{x}^*$.
2: **while** $\mathbf{x}$ is not integral **do**
3:     do one of the two operations arbitrarily:
       Operation 1:
4:        choose one coordinate $a \in [\bar{n}]$ with $x_a \in (0,1)$, two reals $\delta_1 \in (0, x_a]$ and $\delta_2 \in (0, 1 - x_a]$.
5:        with probability $\frac{\delta_2}{\delta_1 + \delta_2}$ **do:** $x_a \leftarrow x_a - \delta_1$, **else do:** $x_a \leftarrow x_a + \delta_2$.
       Operation 2:
6:        choose two distinct coordinates $a, b \in [\bar{n}]$ with $x_a, x_b \in (0,1)$, two reals $\delta_1 \in (0, \min\{x_a, 1 - x_b\}]$, and $\delta_2 \in (0, \min\{1 - x_a, x_b\}]$.
7:        with probability $\frac{\delta_2}{\delta_1 + \delta_2}$ **do:** $x_a \leftarrow x_a - \delta_1, x_b \leftarrow x_b + \delta_1$.
8:             **else do:** $x_a \leftarrow x_a + \delta_2, x_b \leftarrow x_b - \delta_2$.
9: **return x**

We shall use $\mathbf{x}$ to denote the final $\mathbf{x}$ vector returned by Algorithm 1. It can be shown that x_j's are negatively correlated following a similar argument to that used in the pipage rounding procedure by [8]. However, it is unknown whether the negative correlation alone suffices to establish the Chernoff-type concentration bounds we need. Therefore, we provide a direct proof of the concentration bound for $\mathbf{x}$ using the concave estimator technique from [18]. We prove the following theorem in Appendix A.1:

Theorem 2.7. *Let $v : 2^{[\bar{n}]} \to \mathbb{R}_{\geq 0}$ be a monotone submodular function with marginal values at most 1, and $F : [0,1]^{\bar{n}} \to \mathbb{R}_{\geq 0}$ be the multilinear extension of v. Let $\mathbf{x}^* \in [0,1]^{\bar{n}}$, $\mu = F(\mathbf{x}^*)$, $\mathbf{x} \in \{0,1\}^{\bar{n}}$ be the output of Algorithm 1 for the input $\mathbf{x}^*$, and $U = \{i \in [\bar{n}] : x_i = 1\}$. Then, for any $\delta \in (0,1)$, we have*

$$\Pr[v(U) \leq (1-\delta)\mu] \leq e^{-\delta^2 \mu / 2}.$$

3 Iterative Rounding for Weighted Submodular Nash Social Welfare using Configuration LP

In this section, we give our algorithm for the weighted submodular NSW problem. We describe the configuration LP relaxation in Section 3.1, the construction of the bipartite multi-graph G in Section 3.2, and the iterative rounding algorithm in Section 3.3 respectively.

3.1 LP Formulation

We start with the configuration LP relaxation (Conf-LP) for our problem, which is the same as the one used in [11], except now, each v_i is a submodular function. In the correspondent integer program, we have a variable $y_{i,S} \in \{0,1\}$ for every agent $i \in N$ and set $S \subseteq M$ indicating if the set of items assigned to agent i is precisely S or not. The objective is to maximize the logarithm of the weighted Nash social welfare, which is $\sum_{i \in N, S \subseteq M} w_i \cdot y_{i,S} \cdot \ln(v_i(S))$. (1) ensures that each item is assigned to exactly one agent. (2) ensures that each agent is given precisely one item set.

$$\max \quad \sum_{i \in N, S \subseteq M} w_i \cdot y_{i,S} \cdot \ln(v_i(S)) \qquad \text{(Conf-LP)}$$

s.t.

$$\sum_{S:j \in S} \sum_{i \in N} y_{i,S} = 1, \qquad \forall j \in M \qquad (1)$$

$$\sum_{S \subseteq M} y_{i,S} = 1, \qquad \forall i \in N \qquad (2)$$

$$y_{i,S} \geq 0, \qquad \forall i \in N, S \subseteq M \qquad (3)$$

There are an exponential number of variables in the LP. Using standard techniques, we can consider the dual of the LP and design a separation oracle for it. However, unlike the additive valuation case, for which the separation oracle incurs an additive ϵ error, the error becomes $\ln(\frac{e}{e-1} + \epsilon)$ for the submodular valuation case (this corresponds to a multiplicative factor of $(\frac{e}{e-1} + \epsilon)$ for the weighted NSW objective). This comes from the approximation factor for the problem of maximizing a monotone submodular function with a knapsack constraint [33]. The proof of the lemma is deferred to Appendix B.1:

Lemma 3.1. *For any constant $\epsilon > 0$, the Configuration LP (Conf-LP) can be solved in polynomial time within an additive error of $\ln(\frac{e}{e-1} + \epsilon)$.*

3.2 Construction of Bipartite Multi-Graph $G = (N \cup M, E)$ with Marked and Unmarked Edges

After we solve (Conf-LP), we obtain a solution $\mathbf{y}^* := (y_{i,S}^*)_{i \in N, S \subseteq M}$, represented using the list of non-zero coordinates. From now on, we say $S \subseteq M$ is a configuration for an agent i if $y_{i,S}^* > 0$.

Definition 3.2 (Large and Small Items). *For each agent i and configuration S for i, let $\kappa_S^{(i)}$ be the largest item in S, i.e., $\kappa_S^{(i)} := \arg\max_{j \in S} v_i(j)$. We say j is a large item for i if $j = \kappa_S^{(i)}$ for some configuration S for i. Let M_i^{L} be the set of large items for agent i. We say j is a small item for i if $j \in S \setminus \kappa_S^{(i)}$ for some configuration S for i. Let M_i^{S} be the set of small items for i.*

Note that an item may be both a large and small item for an agent i. It may also happen that for a small item j, and a large item j' for an agent i, we have $v_i(j) > v_i(j')$.

Our rounding algorithm is similar to that of [25] for the unrelated machine weighted completion time problem. We build a bipartite multi-graph $G := (N \cup M, E)$ between agents N and items M, where each edge in E is either *marked* or *unmarked*. We also define a vector $\mathbf{x}^*$ over the edges. For each agent i and item $j \in M_i^{\mathrm{L}}$, we create a marked edge ij with x^*-value $x_{ij}^* := \sum_{S:\kappa_S^{(i)}=j} y_{i,S}^*$. For each agent i and item $j \in M_i^{\mathrm{S}}$, we create an unmarked edge with x^*-value $x_{ij}^* := \sum_{S:j \in S \setminus \kappa_S^{(i)}} y_{i,S}^*$. So, there might be two parallel edges ij between an agent i and an item j; in this case, one of them is marked, and the other is unmarked. From now on, when we refer to an edge ij, we assume we know its identity, which will decide if j is large or small.

Notice that the constructed vector $\mathbf{x}^*$ has the following properties:

(i) For any item $j \in M$, we have $\sum_{ij \in E} x_{ij}^* = 1$.

(ii) For any agent $i \in N$, we have $\sum_{j \in M_i^{\mathrm{L}}} x_{ij}^* = 1$.

The first property is due to Constraint (2) of (Conf-LP), and the second is due to Constraint (1) of (Conf-LP). An example can be found in Figure 1.
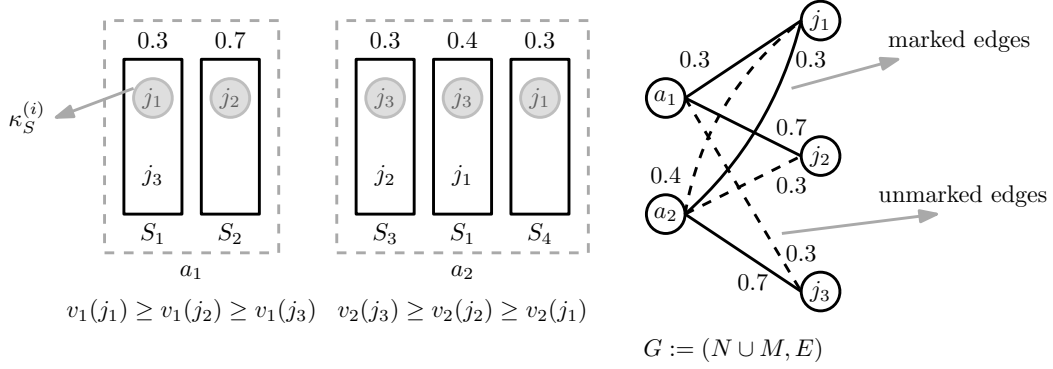
Figure 1: Illustration for the constructed bipartite multi-graph. There are two agents and three items, and the agents' preference for items is shown in the figure. Suppose that, after solving (Conf-LP), we obtain $y^*_{1,S_1} = 0.3, y^*_{1,S_2} = 0.7, y^*_{2,S_3} = 0.3, y^*_{2,S_1} = 0.4, y^*_{2,S_4} = 0.3$. By agents' preference, we know that $M^L_1 = \{j_1, j_2\}$ and $M^L_2 = \{j_1, j_3\}$, they are marked by gray cycle. The small item set is $M^S_1 = \{j_3\}, M^S_2 = \{j_1, j_2\}$. Then, we can create a bipartite multi-graph as stated above. In the right part of the figure, we use a solid line to represent the marked edges (edges between agents and large items) and a dashed line to represent the unmarked edges (edges between agents and small items). The example shows two edges between agent a_2 and item j_1; thus, the bipartite graph can be a multi-graph. Verifying the two properties of the constructed vector $\mathbf{x}^*$ is also easy.

3.3 Rounding x into an Integral Solution

Our iterative rounding algorithm is also similar to that of [25]. During the algorithm, we maintain a vector $\mathbf{x} \in [0,1]^E$, which is set to $\mathbf{x}^*$ initially. In each round, we find either a *marked cycle* or a *pseudo-marked path* in fsupp($\mathbf{x}$), where we shall use fsupp($\mathbf{x}$) := $\{ij \in E : x_{ij} \in (0,1)\}$ to denote the set of edges in E with strictly fractional x values. We apply the rotation/shifting operation to the structure we found until $\mathbf{x}$ becomes integral. The operations we use here are slightly simpler than those in [25] as we maintain the total x-values of marked edges incident to an agent, while the algorithm in [25] needs to maintain the total "volume", which makes the procedure more complicated.

First, we define *marked cycles* and *pseudo-marked paths*. In the remainder of this subsection, we use i and j to represent an agent and an item, respectively.

Definition 3.3 (Marked Cycle and Pseudo-Marked Path). *Given a spanning sub-graph $G' = (N \cup M, E')$ of G, a marked cycle in G' is a simple cycle in G' that consists of marked edges only. A pseudo-marked path $(i_1, j_1, \ldots, j_k, i_{k+1}), k \geq 1$ is a (not-necessarily-simple) path of G' such that*

(3.3a) the subpath $(j_1, \ldots, j_k)$ is a simple path that consists of marked edges only, and

(3.3b) the edges (i_1, j_1) and (j_k, i_{k+1}) are distinct unmarked edges.

For a pseudo-marked path, we do not impose any other conditions that are not stated in the definition. For example, i_1 and/or i_{k+1} may be the same as some agent in $\{i_1, i_2, \ldots, i_{k-1}\}$; in this case, the pseudo path shall contain cycles with both marked and unmarked edges. It may also happen that $i_1 = i_{k+1}$. It is also possible that $k = 1$, in which case the pseudo path consists of two distinct unmarked edges, which implies $i_1 \neq i_2$.

Given any vector $\mathbf{x} \in [0,1]^E$, we define $G(\mathbf{x}) := (V, \text{fsupp}(\mathbf{x}))$ to be the spanning graph of $\mathbf{x}$ containing the edges with x-values strictly between 0 and 1. The iterative rounding algorithm is described in Algorithm 2.

We first assume that in Step 3, we can always find a marked cycle or a pseudo-marked path. Then, it is easy to see that we maintained the following two invariants

(I1) $\sum_{ij \in E} x_{ij} = 1$ for every item $j \in M$.

8

Algorithm 2 Iterative Rounding Algorithm for $\mathbf{x}^*$

Input: the bipartite multi-graph $G := (N \cup M, E)$ and the fractional vector $\mathbf{x}^* := (x_e^*)_{e \in E}$.

Output: an integral solution $\mathbf{x}$.

1: $\mathbf{x} \leftarrow \mathbf{x}^*$.

2: **while** there exists a variable x_e such that $0 < x_e < 1$ **do**

3: find either a marked cycle $(i_1, j_1, i_2, j_2, \ldots, i_k, j_k, i_{k+1} = i_1)$ or a pseudo-marked path $(i_1, j_1, i_2, j_2, \ldots, i_k, j_k, i_{k+1})$ in $G(\mathbf{x})$.

4: $\delta_1 \leftarrow \min\left\{ \min_{a \in [k]} x_{i_a j_a}, \min_{a \in [k]} (1 - x_{j_a i_{a+1}}) \right\}, \qquad \delta_2 \leftarrow \min\left\{ \min_{a \in [k]} (1 - x_{i_a j_a}), \min_{a \in [k]} x_{j_a i_{a+1}} \right\}.$

5: with probability $\dfrac{\delta_2}{\delta_1 + \delta_2}$ **do**: **for** every $a \in [k]$ **do**: $x_{i_a j_a} \leftarrow x_{i_a j_a} - \delta_1, x_{j_a i_{a+1}} \leftarrow x_{j_a i_{a+1}} + \delta_1$.

6:    else **do**: **for** every $a \in [k]$ **do**: $x_{i_a j_a} \leftarrow x_{i_a j_a} + \delta_2, x_{j_a i_{a+1}} \leftarrow x_{j_a i_{a+1}} - \delta_2$.

7: **return** $\mathbf{x}$.

(I2) $\sum_{j \in M_i^{\mathrm{L}}} x_{ij} = 1$ for every agent $i \in N$.

Then, the algorithm terminates in polynomial time as the x-value of at least one edge becomes integral in each iteration.

It remains to show that the goal in Step 3 can be achieved:

Lemma 3.4. *Let $\mathbf{x}$ be the vector before Step 3 in some iteration of the while loop; so $\mathbf{x}$ is not integral. Then, $G(\mathbf{x})$ contains a marked cycle or a pseudo-marked path.*

Proof. If the marked edges in $G(\mathbf{x})$ contain a cycle, then we are done. Assume this does not happen; so the marked edges form a forest.

If there is at least one marked edge in $G(\mathbf{x})$, then we can take a non-empty path of marked edges between two leaf vertices in the forest. Due to Invariant (I2), both leaves must be items. So, let $j_1, i_2, j_2, i_3, \ldots, i_k, j_k$ be the non-empty simple path of marked edges; j_1 and j_k are leaves in the forest. Due to Invariant (I1), there must be a marked edge incident to j_1 in $G(\mathbf{x})$; the same holds for j_k. So, concatenating the two marked edges and the path $j_1, i_2, j_2, i_3, \ldots, i_k, j_k$ gives us a pseudo-marked path.

Now suppose there are no marked edges in $G(\mathbf{x})$. As $\mathbf{x}$ is not integral, there is some item j which is not integrally assigned in $\mathbf{x}$. Then j must be incident to two distinct unmarked edges due to Invariant (I1). The two unmarked edges form a pseudo-marked path. $\qquad\square$

For every agent i, let $\mathbf{x}^{i,\mathrm{umk}}$ be the vector $\mathbf{x}$ restricted to the set of unmarked edges incident to i. If we only focus on how $\mathbf{x}^{i,\mathrm{umk}}$ changes, then Algorithm 2 falls into the algorithmic template stated in Algorithm 1. In particular, if in an iteration we rotate a marked cycle, then $\mathbf{x}^{i,\mathrm{umk}}$ is unchanged. Suppose we shift a pseudo-marked-path $(i_1, j_1, \ldots, i_k, j_k, i_{k+1})$ in an iteration. If $i \notin \{i_1, i_{k+1}\}$, then $\mathbf{x}^{i,\mathrm{umk}}$ is unchanged; if $i = i_1, i \neq i_{k+1}$ or $i \neq i_1, i = i_{k+1}$, then Operation 1 is performed in the iteration in Algorithm 2; if $i = i_1 = i_{k+1}$, then Operation 2 is performed. Thus, we can apply the concentration bound over $\mathbf{x}^{i,\mathrm{umk}}$ using Theorem 2.7.

4 Analysis of the Approximation Ratio

In this section, we analyze the approximation ratio achieved by the algorithm. For every $e \in E$, let $X_e \in \{0, 1\}$ be the value of x_e returned by Algorithm 2. We first summarize some of the properties we have for the random variables X_e's.

Observation 4.1. *The following properties hold:*

(4.1a) *For each $e \in E$, we have $\mathbb{E}[X_e] = x_e^*$.*

(4.1b) For each item j, we have $\sum_{ij \in E} X_{ij} = 1$ with probability 1.

(4.1c) For each agent i, we have $\sum_{j \in M_i^L} X_{ij} = 1$ with probability 1.

Proof. Fix an edge $e \in E$. Focus on an iteration of the loop in Algorithm 2. Let x_e^{old} and x_e^{new} be the value of x_e at the beginning and end of the iteration respectively. Then it is easy to see that $\mathbb{E}[x_e^{\text{new}}|x_e^{\text{old}}] = x_e^{\text{old}}$. Property (4.1a) then follows. Property (4.1b) and (4.1c) follow from Invariant (I1) and (I2), respectively. $\square$

Till the end of the section, we fix an agent $i \in N$ and let T be the set of items assigned to i by our algorithm. As i is fixed, we omit subscripts i from most of the notations: we use $v(S), \kappa(S), y_S^*, M^L$ and M^S to denote $v_i(S), \kappa_i(S), y_{i,S}^*, M_i^L$ and M_i^S respectively. Notice that by (4.1c) of Observation 4.1, T contains exactly one large item, i.e., an item in M^L. We use k^L to denote the unique large item in T. Let $T^S := T \cap M^S$ be the set of small items assigned to the agent i. So, $T = \{k^L\} \uplus T^S$. The goal of the section is to prove Lemma 4.2.

Lemma 4.2. $\mathbb{E}\left[\ln(v(T))\right] \geq \sum_S y_S^* \ln v(S) - \left(3 + \dfrac{26}{e^3} + \ln 2\right)$.

We show how the lemma implies Theorem 1.1.

Proof of Theorem 1.1. In this proof, we consider all agents $i \in N$ and thus we include the subscript i in all notations. Also, let T_i denote the set of items assigned to the agent i by our algorithm. We apply Lemma 4.2 to each agent $i \in N$, and by the linearity of expectation and $\sum_{i \in N} w_i = 1$, we have:

$$\mathbb{E}\left[\sum_{i \in N} w_i \cdot \ln v_i(T_i)\right] \geq \sum_{i \in N, S \subseteq M} w_i \cdot y_{i,S}^* \ln v_i(S) - \left(3 + \frac{26}{e^3} + \ln 2\right).$$

Raising both sides to the exponent and applying Jensen's inequality, we have:

$$\mathbb{E}\left[\prod_{i \in N} v_i(T_i)^{w_i}\right] \geq \exp\left(-\left(3 + \frac{26}{e^3} + \ln 2\right)\right) \cdot \exp\left(\sum_{i \in N, S \subseteq M} w_i \cdot y_{i,S}^* \ln v_i(S)\right).$$

By Lemma 3.1, we have:

$$\exp\left(\sum_{i \in N, S \subseteq M} w_i \cdot y_{i,S}^* \ln v_i(S)\right) \geq \text{OPT} \cdot \left(1 - \frac{1}{e} - \epsilon\right).$$

Thus, we get:

$$\mathbb{E}\left[\prod_{i \in N} v_i(T_i)^{w_i}\right] \geq \exp\left(-\left(3 + \frac{26}{e^3} + \ln 2\right)\right) \cdot \left(1 - \frac{1}{e} - \epsilon\right) \cdot \text{OPT}.$$

Therefore, the overall approximation factor is $\left(\frac{e}{e-1} + \epsilon\right) \cdot 2 \cdot \exp(3 + 26/e^3) < 233 + \epsilon$. $\square$

Now, it remains to show Lemma 4.2. To this end, we first find a lower and upper bound for the value of $\mathbb{E}[\ln v(T)]$ and $\sum_S y_S^* \ln v(S)$; see Section 4.1. Then, we shall compare these two bounds in Section 4.2 to obtain Lemma 4.2.

4.1 Input and Output Distributions

We have two distributions over sets of items: the *input distribution*, which is the distribution of configurations for i, and the *output distribution*, which corresponds to the distribution for the set of items assigned to agent i by Algorithm 2. We set up notations for both distributions.

Notations and Histogram for Input Distribution. In the input distribution, the probability for a set S is y_S^*; recall that a configuration S for i is a set S with $y_S^* > 0$. So the total probabilities over all configurations S is 1 by the LP constraint (2). We define a function $\pi : [0,1] \to 2^M$ as follows. We sort all the configurations S in descending order of $\kappa(S)$, breaking ties arbitrarily. Let $\prec$ denote this order: $S' \prec S$ if S' appears before S in the order, and $S' \preceq S$ if and only if $S' \prec S$ or $S' = S$. Then, π is defined as follows: $\pi(t)$ for every $t \in (0,1]$ is the first set S in the order such that $\sum_{S' \preceq S} y_{S'}^* \geq t$. Let $\pi(0)$ be the first S in the order with $y_S^* > 0$. For notational convenience, we define $\kappa_t := \kappa(\pi(t))$ for every $t \in [0,1]$. Let $u_t = v(\kappa_t)$ and $B_t = v(\pi_t \setminus \kappa_t)$.

It is convenient to visualize the notations using a histogram, denoted in the subfigure (i) of Figure 2. We create a rectangle for each configuration S with width $y_S^* > 0$ and height $u_t + B_t = v(\kappa_t) + v(\pi_t \setminus \kappa_t) \geq v(\pi_t)$. The bottom portion of the rectangle of height u_t is colored dark, and the top portion of height B_t is colored white. The rectangles are arranged horizontally above the interval $[0,1]$ according to the order $\prec$.

Notations and Histograms for Output Distribution. Notice that the agent i is assigned exactly one large item by Property (4.1c) of Observation 4.1. So our main focus is on the set T^S of small items assigned to i. For each $t \in (0,1]$, we define C_t as the largest number A such that $\Pr[v(T^S) \geq A] \geq x$. Let $C_0 = \lim_{t \to 0^+} C_t$.

We can create two histograms for the output distribution, one for the large items, and the other for small items. See the subfigure (ii) of Figure 2. Due to Property (4.1a) of Observation 4.1, the histogram for large items is the same as the dark portion of the histogram for the input distribution: the height at position t is u_t. So, for each set U that T^S can take, we create a rectangle of height $v(U)$ and width $\Pr[T^S = U]$, and the rectangles are sorted in descending order of $v(U)$ values from left to the right in the histogram. In the histogram for small items, the height at position t is C_t.

We remark that we disregard the correlation between the large item and small items assigned to i, by creating two separate histograms. In both histograms, we sort the rectangles according to the heights. So, the large item and the small item set at position t in their respective histogram may not occur simultaneously.
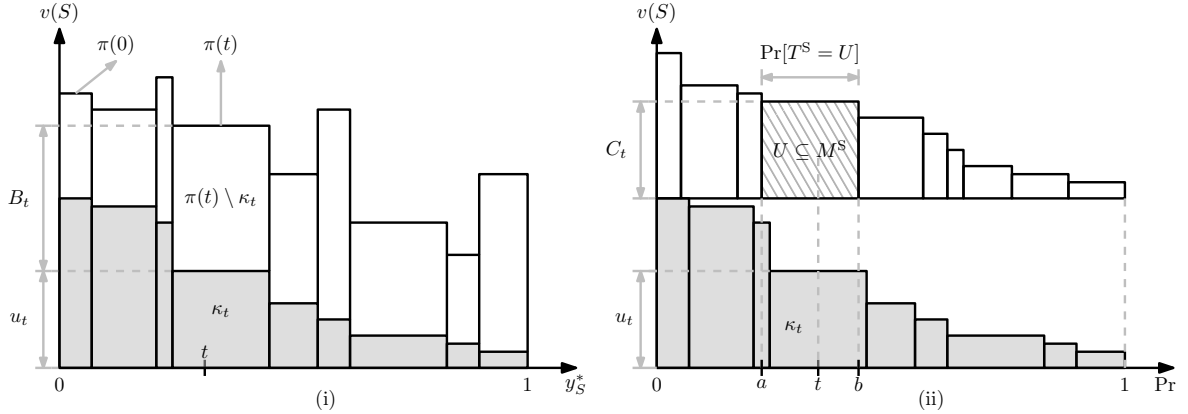

Figure 2: Illustration for the $\pi(\cdot)$ function and C_t, which are shown in the subfigure (i) and (ii), respectively. For simplicity, the figure considers the case when the valuation function v is additive. In the subfigure (i), for each set $S \subseteq M$ with $y_S^* > 0$, we have a rectangle. We sort these rectangles in non-increasing order according to the value of the largest item in each set. In each rectangle, the gray part is the size of the largest item, and the remaining part is the small item part. In the subfigure (ii), for each possible algorithm's output of small items, we have a rectangle. We also sort these rectangles in non-increasing order according to the value of each set.

With the notations set up, we can relate the two quantities in Lemma 4.2 with the quantities we introduced:

Lemma 4.3. $\sum_S y_S^* \ln v(S) \leq \int_0^1 \ln(u_t + B_t) \, dt$.

Proof. Consider any $t \in (0,1)$, and due to the subadditive property of the valuation function, we have $v(\pi(t)) \leq u_t + B_t$. So, $\int_0^1 \ln v(\pi(t)) \, dt \leq \int_0^1 \ln(u_t + B_t) \, dt$. Observe that $\sum_S y_S^* \ln v(S) = \int_0^1 \ln v(\pi(t)) \, dt$, as desired. We remark that the inequality above becomes an equality when v is additive. $\qquad\square$

Lemma 4.4. $\mathbb{E}[\ln v(T)] \geq \int_0^1 \ln(u_t + C_t) \, dt - \ln 2$.

Proof. Before proving the lemma, we first show the useful properties of two sequences.

Claim 4.5. *Consider two non-increasing nonnegative sequences $\{a_j\}_{j=1}^\ell$ and $\{b_j\}_{j=1}^\ell$, then for any permutation σ on $[\ell]$, we have*

$$\sum_{j=1}^\ell \ln(a_j + b_{\sigma(j)}) \geq \sum_{j=1}^\ell \ln(a_j + b_j),$$

where $\sigma(j)$ is the index at the j-th position of permutation σ.

The Claim 4.5 can be proved via the exchange argument; a proof can be found in [30, Theorem 2]. Now, suppose that the Claim 4.5 is correct. We have $v(T) \geq v(k^L)$ and $v(T) \geq v(T^S)$ by the monotonicity of v, so it suffices to show that

$$\mathbb{E}[\ln(v(k^L) + v(T^S))] \geq \int_0^1 \ln(u_t + C_t) \, dt. \qquad (4)$$

To see that, we assume $\Pr[T^S = \bar{T}, k^L = \bar{k}]$ for any $\bar{T} \subseteq M^S$ and $\bar{k} \in M^L$ is an integer multiple of Δ, for a sufficiently large integer $\Delta > 0$. We construct two multi-sets Z^L and Z^S of Δ integers as follows. For every pair $(\bar{T}, \bar{k})$ with positive $\Pr[T^S = \bar{T}, k^L = \bar{k}]$, we add $\Delta \Pr[T^S = \bar{T}, k^L = \bar{k}]$ copies of $v(\bar{T})$ to Z^L, and $\Delta \Pr[T^S = \bar{T}, k^L = \bar{k}]$ copies of $v(\bar{k})$ to Z^S. Let the sequences $(a_1, a_2, \ldots, a_\Delta)$ and $(b_1, b_2, \ldots, b_\Delta)$ respectively be the sets Z^L and Z^S sorted in non-increasing order. After scaling up by Δ, the left-side of (4) becomes $\sum_{j=1}^\Delta \ln(a_j + b_{\sigma(j)})$ for some permutation σ of $[\Delta]$, and the right-side becomes $\sum_{j=1}^\Delta \ln(a_j + b_j)$. (4) then follows from Claim 4.5. $\qquad\square$

With the two lemmas, it remains for us to compare $\int_0^1 \ln(u_t + B_t) \, dt$ with $\int_0^1 \ln(u_t + C_t) \, dt$.

4.2 Comparing $\int_0^1 \ln(u_t + B_t) \, dt$ and $\int_0^1 \ln(u_t + C_t) \, dt$

By (2.3b) of Lemma 2.3, we know that the truncated function f is a lower bound of v. We shall use the lower bound f in our analysis as the truncation gives an upper bound on the values of individual items, which allows us to apply the Chernoff-type concentration bound stated in Theorem 2.7. The following helper lemma will be useful later, which suggests a lower bound of the small items' expected value under the truncated valuation function.

Lemma 4.6. *Let $\rho \in (0,1)$, let $f := v^{(u_\rho + C_\rho)}$ be the function obtained from v by truncating individual values by $u_\rho + C_\rho$, as in Definition 2.2. Then we have*

$$\int_0^\rho f(\pi(t) \setminus \kappa_t) \, dt \geq (u_\rho + C_\rho) \cdot \int_0^\rho \frac{B_t}{u_t + C_t} \, dt.$$

Proof. We focus on any $t \in (0, \rho)$, and we have $\max_{j \in \pi(t) \setminus \kappa_t} v(j) \leq u_t \leq u_t + C_t$. Thus, by (2.3b) of Lemma 2.3, we have

$$f(\pi(t) \setminus \kappa_t) \geq \frac{u_\rho + C_\rho}{u_t + C_t} \cdot v(\pi(t) \setminus \kappa_t) = \frac{B_t}{u_t + C_t} \cdot (u_\rho + C_\rho).$$

Therefore, we have

$$\int_0^\rho f(\pi(t) \setminus \kappa_t) \, \mathrm{d}t \geq \int_0^\rho \frac{B_t}{u_t + C_t} \cdot (u_\rho + C_\rho) \, \mathrm{d}t = (u_\rho + C_\rho) \cdot \int_0^\rho \frac{B_t}{u_t + C_t} \, \mathrm{d}t. \qquad \square$$

We then prove the following key lemma:

Lemma 4.7. *For every $\lambda \geq 0$, we have*

$$\int_0^1 \mathbf{1}\{\ln B_t > \ln(u_t + C_t) + \lambda\} \, \mathrm{d}t \leq (6\lambda + 1)\mathrm{e}^{-\lambda}.$$

Proof. We can assume that $\lambda \geq 1$ since otherwise $(6\lambda + 1)\mathrm{e}^{-\lambda} > 1$ and the lemma holds trivially. For each $t \in (0, 1)$, we define $\theta_t = \mathbf{1}\{\ln B_t > \ln(u_t + C_t) + \lambda\} \in \{0, 1\}$, i.e., $\theta_t = 1$ if $\ln B_t > \ln(u_t + C_t) + \lambda$; otherwise, $\theta_t = 0$. It is sufficient to prove $\int_0^{1-\mathrm{e}^{-\lambda}} \theta_t \, \mathrm{d}t \leq 6\lambda \mathrm{e}^{-\lambda}$. Assume towards the contradiction that $\int_0^{1-\mathrm{e}^{-\lambda}} \theta_t \, \mathrm{d}t > 6\lambda \mathrm{e}^{-\lambda}$. Then, we can find a $z \in (0, 1-\mathrm{e}^{-\lambda})$ such that $\int_0^z \theta_t \, \mathrm{d}t = 6\lambda \mathrm{e}^{-\lambda}$. Note that $\theta_z = 1$; so, we have $\frac{B_z}{u_z + C_z} > \mathrm{e}^\lambda$.

Now, we define the truncated function f using the value of $u_z + C_z$, i.e., $f := v^{(u_z + C_z)}$. We aim to use the concentration bound proved in Theorem 2.7, which bounds the value of small items deviating from its multilinear extension.

Let $\mathbf{x}^{\mathrm{umk}} := (\mathbf{x}_{ij}^*)_{j \in M_i^{\mathrm{S}}}$ be fractional assignment for small items to agent i. By Lemma 2.3, we know that f is a monotone submodular function. Let $F(\cdot)$ be the multilinear extension of the function f. Our goal is to lower bound the value of $F(\mathbf{x}^{\mathrm{umk}})$ so that we can use the concentration bound. Let $f^+ : [0, 1]^{M^{\mathrm{S}}} \to \mathbb{R}_{\geq 0}$ be the concave extension of f, defined in Definition 2.5.

Now, we consider the small item part of the (Conf-LP)'s solution $(y_S^*)_{S \subseteq M^{\mathrm{S}}}$. By the definition of $\mathbf{x}^{\mathrm{umk}}$ and (Conf-LP)'s constraints, $\mathbf{x}^{\mathrm{umk}}$ and $(y_S^*)_{S \subseteq M^{\mathrm{S}}}$ must satisfy all constraints stated in the concave extension in Definition 2.5, i.e., (i) $\sum_{S \subseteq M^{\mathrm{S}}} y_S^* \leq 1$ since (2) of (Conf-LP); (ii) for all small item set $S \subseteq M^{\mathrm{S}}$, $y_S^* \geq 0$; (iii) for each small item $j \in M^{\mathrm{S}}$, $\sum_{S:j \in S} y_S^* \leq x_j^{\mathrm{umk}}$. Thus, we know that $f^+(\mathbf{x}^{\mathrm{umk}})$ is at least as large as the expected value of small items in the LP solution, i.e.,

$$f^+(\mathbf{x}^{\mathrm{umk}}) \geq \int_0^1 v(\pi(t) \setminus \kappa_t) \, \mathrm{d}t \geq \int_0^1 f(\pi(t) \setminus \kappa_t) \, \mathrm{d}t \geq \int_0^z \theta_t \cdot f(\pi(t) \setminus \kappa_t) \, \mathrm{d}t.$$

Applying Lemma 4.6 by setting ρ as z, we have:

$$\int_0^z \theta_t \cdot f(\pi(t) \setminus \kappa_t) \, \mathrm{d}t \geq (u_z + C_z) \cdot \int_0^z \theta_t \cdot \frac{B_t}{u_t + C_t} \, \mathrm{d}t \geq (u_z + C_z) \cdot \mathrm{e}^\lambda \cdot \int_0^z \theta_t \, \mathrm{d}t = (u_z + C_z) \cdot 6\lambda.$$

By Lemma 2.6, we have

$$F(\mathbf{x}^{\mathrm{umk}}) \geq \left(1 - \frac{1}{\mathrm{e}}\right) f^+(\mathbf{x}^{\mathrm{umk}}) \geq (u_z + C_z) \cdot 6\left(1 - \frac{1}{\mathrm{e}}\right)\lambda.$$

Notice that $f(\cdot)$ is a submodular function such that the maximum value of each single item is at most $u_z + C_z$. Thus, the marginal value of $\frac{f(\cdot)}{u_z + C_z}$ is in $[0, 1]$. Thus, the concentration bound stated in

Theorem 2.7 works for $\frac{f(\cdot)}{u_z+C_z}$. Then, we have the following inequalities:

$$\Pr[v(T^{\mathrm{S}}) < \lambda(u_z + C_z)]$$

$$\leq \Pr[f(T^{\mathrm{S}}) < \lambda(u_z + C_z)] \qquad\qquad [f(T^{\mathrm{S}}) \leq v(T^{\mathrm{S}})]$$

$$= \Pr\left[\frac{f(T^{\mathrm{S}})}{u_z + C_z} < \lambda\right]$$

$$= \Pr\left[\frac{f(T^{\mathrm{S}})}{u_z + C_z} < \left(1 - \frac{5\mathrm{e}-6}{6(\mathrm{e}-1)}\right)\left(6(1-\frac{1}{\mathrm{e}})\lambda\right)\right]$$

$$\leq \exp\left(-\frac{(\frac{5\mathrm{e}-6}{6(\mathrm{e}-1)})^2 \cdot 6(1-\frac{1}{\mathrm{e}})\lambda}{2}\right) \qquad\qquad [\text{Theorem 2.7 with } \delta = \frac{5\mathrm{e}-6}{6(\mathrm{e}-1)}, \mu \geq 6(1-\frac{1}{\mathrm{e}})\lambda]$$

$$\leq \mathrm{e}^{-\lambda}.$$

Thus, with probability at least $1-\mathrm{e}^{-\lambda}$, we have $v(T^{\mathrm{S}}) \geq \lambda(u_z+C_z) > u_z+C_z$ since $\lambda \geq 1$. We use t^* to represent $1 - \mathrm{e}^{-\lambda}$. Recall that the definition of C_t is the largest A such that $\Pr[v(T^{\mathrm{S}}) \geq A] \geq t$ holds. Therefore, we have $C_{t^*} > u_z + C_z \geq C_z$. Notice that $z < 1 - \mathrm{e}^{-\lambda} = t^*$. This contradicts the fact that C_t is non-increasing over t. $\qquad\square$

Now, we are ready to bound the gap between the expected value of the LP solution and the solution outputted by Algorithm 2.

Lemma 4.8. $\int_0^1 \left(\ln(u_t + B_t) - \ln(u_t + C_t)\right) \mathrm{d}t \leq 3 + \frac{26}{\mathrm{e}^3}.$

Proof. We have the following inequality:

$$\int_0^1 (\ln(u_t + B_t) - \ln(u_t + C_t))\,\mathrm{d}t$$

$$\leq \int_0^\infty \Pr\left[\frac{u_t + B_t}{u_t + C_t} > \mathrm{e}^\lambda\right]\mathrm{d}\lambda \quad \leq \quad \int_0^\infty \Pr\left[\frac{B_t}{u_t + C_t} > \mathrm{e}^\lambda - 1\right]\mathrm{d}\lambda \qquad\qquad [u_t \leq u_t + C_t]$$

$$\leq 3 + \int_3^\infty \Pr\left[\frac{B_t}{u_t + C_t} > \mathrm{e}^\lambda - 1\right]\mathrm{d}\lambda \quad \leq \quad 3 + \int_3^\infty \Pr\left[\frac{B_t}{u_t + C_t} > \mathrm{e}^{\lambda - 1/\mathrm{e}^3}\right]\mathrm{d}\lambda$$

$$= 3 + \int_{3-1/\mathrm{e}^3}^\infty \Pr\left[\frac{B_t}{u_t + C_t} > \mathrm{e}^\lambda\right]\mathrm{d}\lambda \quad \leq \quad 3 + 1/\mathrm{e}^3 + \int_3^\infty \Pr\left[\frac{B_t}{u_t + C_t} > \mathrm{e}^\lambda\right]\mathrm{d}\lambda$$

$$\leq 3 + 1/\mathrm{e}^3 + \int_3^\infty (6\lambda + 1)\mathrm{e}^{-\lambda}\,\mathrm{d}\lambda \qquad\qquad [\text{by Lemma 4.7}]$$

$$= 3 + \frac{26}{\mathrm{e}^3}. \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$$

Now, we are ready to show Lemma 4.2.

Proof of Lemma 4.2. By Lemma 4.3, Lemma 4.4, and Lemma 4.8, we immediately obtain the following inequalities:

$$\mathbb{E}[\ln v(T)] - \sum_S y_S^* \ln v(S) \geq \int_0^1 \ln(u_t + C_t)\,\mathrm{d}t - \int_0^1 \ln(u_t + B_t)\,\mathrm{d}t - \ln 2$$

$$\geq -\left(3 + \frac{26}{\mathrm{e}^3} + \ln 2\right).$$

This finishes the proof of Lemma 4.2. $\qquad\square$

5   Conclusion

We study the problem of maximizing the weighted Nash social welfare under submodular valuations and give a $(233 + \epsilon)$-approximate algorithm for the problem, which is the first constant approximation for the problem. It improves upon the previous best-known result of $O(nw_{\max})$. Our work leaves several interesting future directions.

It would be interesting to improve the approximation ratio to a small constant. The current best approximation ratio for unweighted Nash social welfare with submodular valuations is $(4 + \epsilon)$ approximation [13]. There is a huge gap between the weighted and unweighted cases.

It is also interesting to consider the weighted NSW problem for more general valuations, such as XOS or subadditive valuations, where the algorithm is assumed to have access to demand oracles to these valuations. For the weighted case with additive valuation, an $(e^{1/e} + \epsilon)$-approximation is known to exist [11], matching the best-known ratio for the unweighted case. This work extends the constant-factor approximation to submodular valuations. Extending this result further beyond the submodular case remains an intriguing open question.

References

[1] Julius Barbanel and Alan Taylor. The Geometry of Efficient Fair Division. *The Geometry of Efficient Fair Division*, pages 1–462, 01 2005.

[2] Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Finding Fair and Efficient Allocations. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, EC '18, page 557–574, New York, NY, USA, 2018. Association for Computing Machinery.

[3] Steven J. Brams and Alan D. Taylor. *Fair Division: From Cake-Cutting to Dispute Resolution.* Cambridge University Press, 1996.

[4] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. *Handbook of Computational Social Choice.* Cambridge University Press, USA, 1st edition, 2016.

[5] Adam Brown, Aditi Laddha, Madhusudhan Reddy Pittu, and Mohit Singh. Approximation Algorithms for the Weighted Nash Social Welfare via Convex and Non-Convex Programs. In *Proceedings of the Thirty-Fifth ACM-SIAM Symposium on Discrete Algorithms*, 2024.

[6] Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[7] Suchan Chae and Hervé Moulin. Bargaining Among Groups: An Axiomatic Viewpoint. *International Journal of Game Theory*, 39(1):71–88, 2010.

[8] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 575–584. IEEE, 2010.

[9] Dagmawi Mulugeta Degefu, Weijun He, Liang Yuan, An Min, and Qi Zhang. Bankruptcy to surplus: Sharing transboundary river basin's water under scarcity. *Water Resources Management*, 32:2735–2751, 2018.

[10] Shahar Dobzinski, Wenzheng Li, Aviad Rubinstein, and Jan Vondrák. A constant-factor approximation for nash social welfare with subadditive valuations. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 467–478, 2024.

[11] Yuda Feng and Shi Li. A note on approximating weighted nash social welfare with additive valuations. In *ICALP*, volume 297 of *LIPIcs*, pages 63:1–63:9. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[12] Jugal Garg, Martin Hoefer, and Kurt Mehlhorn. Satiation in Fisher Markets and Approximation of Nash Social Welfare. *Mathematics of Operations Research*, 0(0):null, 2023.

[13] Jugal Garg, Edin Husić, Wenzheng Li, László A. Végh, and Jan Vondrák. Approximating Nash Social Welfare by Matching and Local Search. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, page 1298–1310, New York, NY, USA, 2023. Association for Computing Machinery.

[14] 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, page 1412–1425, New York, NY, USA, 2021. Association for Computing Machinery.

[15] Jugal Garg, Edin Husić, Wenzheng Li, László A. Végh, and Jan Vondrák. Approximating nash social welfare by matching and local search, 2024, https://arxiv.org/abs/2211.03883.

[16] Jugal Garg, Pooja Kulkarni, and Rucha Kulkarni. Approximating Nash Social Welfare under Submodular Valuations through (Un)Matchings. *ACM Trans. Algorithms*, 19(4), sep 2023.

[17] John C. Harsanyi and Reinhard Selten. A generalized nash solution for two-person bargaining games with incomplete information. *Management Science*, 18(5):P80–P106, 1972.

[18] Nicholas JA Harvey and Neil Olver. Pipage rounding, pessimistic estimators and matrix concentration. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 926–945. SIAM, 2014.

[19] Harold Houba, Gerard van der Laan, and Yuyu Zeng. Asymmetric nash solutions in the river sharing problem. 2013, 2013.

[20] Asei Inoue and Yusuke Kobayashi. An additive approximation scheme for the nash social welfare maximization with identical additive valuations. In *Combinatorial Algorithms: 33rd International Workshop, IWOCA 2022, Trier, Germany, June 7–9, 2022, Proceedings*, page 341–354, Berlin, Heidelberg, 2022. Springer-Verlag.

[21] E. Kalai. Nonsymmetric Nash Solutions and Replications of 2-Person Bargaining. *Int. J. Game Theory*, 6(3):129–133, sep 1977.

[22] Mamoru Kaneko and Kenjiro Nakamura. The Nash Social Welfare Function. *Econometrica*, 47(2):423–435, 1979.

[23] Frank Kelly. Charging and Rate Control for Elastic Traffic. *European Transactions on Telecommunications*, 8(1):33–37, 1997.

[24] Annick Laruelle and Federico Valenciano. Bargaining in Committees as An Extension of Nash's Bargaining Theory. *Journal of Economic Theory*, 132(1):291–305, 2007.

[25] Shi Li. Approximating unrelated machine weighted completion time using iterative rounding and computer assisted proofs. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete algorithms (SODA 2025)*, 2025.

[26] Wenzheng Li and Jan Vondrák. A Constant-Factor Approximation Algorithm for Nash Social Welfare with Submodular Valuations. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 25–36, Los Alamitos, CA, USA, feb 2022. IEEE Computer Society.

[27] Herve Moulin. *Fair Division and Collective Welfare*. The MIT Press, 2004.

[28] John F. Nash. The Bargaining Problem. *Econometrica*, 18(2):155–162, 1950.

[29] 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.

[30] A Oppenheim. Inequalities connected with definite hermitian forms, ii. *The American Mathematical Monthly*, 61(7P1):463–466, 1954.

[31] Jack Robertson and William Webb. *Cake-Cutting Algorithms: Be Fair if You Can*. A K Peters/CRC Press, 1998.

[32] Jrg Rothe. *Economics and Computation: An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[33] Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41–43, 2004.

[34] William Thomson. Replication Invariance of Bargaining Solutions. *Int. J. Game Theory*, 15(1):59–63, mar 1986.

[35] Hal R Varian. Equity, Envy, and Efficiency. *Journal of Economic Theory*, 9(1):63–91, 1974.

[36] Jan Vondrák. Submodularity in combinatorial optimization. *PhD Thesis*, 2007.

[37] H. Peyton Young. *Equity: In Theory and Practice.* Princeton University Press, 1994.

[38] S Yu, EC van Ierland, H-P Weikard, and X Zhu. Nash bargaining solutions for international climate agreements under different sets of bargaining weights. *International Environmental Agreements: Politics, Law and Economics*, 17:709–729, 2017.

A   Missed Proofs from Section 2

A.1   Proof of Theorem 2.7

In this section, we prove Theorem 2.7, which shows that the randomized variables produced by Algorithm 1 satisfy the Chernoff-type concentration bound. We shall utilize the *concave pessimistic estimator* technique developed in [18]. Let $\mathbf{x}^* \in [0,1]^{\bar{n}}$, $v : 2^{[\bar{n}]} \to \mathbb{R}_{\geq 0}$, $F : [0,1]^{\bar{n}} \to \mathbb{R}_{\geq 0}$, $\mu = F(\mathbf{x}^*)$, $\mathbf{x}$, U and $\delta \in (0,1)$ be as defined in the theorem.

We shall use the same concave pessimistic estimator as [18]:

$$g_{t,\theta}(\mathbf{x}) := \mathsf{e}^{-\theta t} \cdot \mathbb{E}_{V \sim \mathcal{D}(\mathbf{x})} \left[\mathsf{e}^{\theta v(V)} \right], \forall \mathbf{x} \in [0,1]^{\bar{n}},$$

where $t := (1-\delta)\mu, \theta := \ln(1-\delta) < 0$ and $\mathcal{D}(\mathbf{x})$ denotes the product distribution over $2^{[\bar{n}]}$ where $\Pr_{V \sim \mathcal{D}(\mathbf{x})}[i \in V] = x_i$ for every $i \in [\bar{n}]$.

Lemma A.1 ([18])**.** *The concave pessimistic estimator has the following properties:*

(A.1a) *for every integral $\mathbf{x} \in \{0,1\}^{[\bar{n}]}$ and the set $U = \{i \in [\bar{n}] : x_i = 1\}$, we have $\mathbf{1}\{v(V) \leq t\} \leq g_{t,\theta}(\mathbf{x})$;*

(A.1b) $g_{t,\theta}(\mathbf{x}^*) \leq \exp(-\delta^2 \mu/2)$;

(A.1c) *the function $g_{t,\theta}$ is concave on the direction $(\mathbf{e}_a - \mathbf{e}_b)$ for any $a, b \in [\bar{n}]$;*

(A.1d) *the function $g_{t,\theta}$ is linear on the direction $\mathbf{e}_a$ for any $a \in [\bar{n}]$.*

One can imagine that $g_{t,\theta}$ estimates the probability that a bad event ($v(U) \leq t$) happens. We remark that (A.1a) of Lemma A.1 is followed by the definition of the concave pessimistic estimator. The (A.1b) of Lemma A.1 is the standard proof for Chernoff bound. In [18, Claim B.1.], they showed that $h(V) := e^{\theta f(V)}, \forall V \subseteq [\bar{n}]$ is a supermodular function if $f : 2^V \to \mathbb{R}$ is submodular. Thus, $g_{t,\theta}$ the multilinear extension of the supermodular function $e^{\theta v}$. It has been shown in [6] that $F(\cdot)$ is convex on the direction $(\mathbf{e}_a - \mathbf{e}_b)$ if F is a multilinear extension of a submodular function. Thus, (A.1c) holds by noting that the negation of a supermodular function is submodular. Since $g_{t,\theta}(\cdot)$ is a multilinear extension, $g_{t,\theta}(\cdot)$ is linear on any direction $\mathbf{e}_a$. This proves (A.1d).

Now, we show that our rounding algorithm has a nice property (Lemma A.2), which is crucial for Theorem 2.7. Lemma A.2 suggests that the probability of bad events (the randomized set has a low value) occurring does not increase as Algorithm 1 runs.

Lemma A.2. *Given any function $f : [0,1]^{\bar{n}} \to \mathbb{R}_{\geq 0}$ such that (i) f is concave on the direction $(\mathbf{e}_a - \mathbf{e}_b)$ for any $a, b \in [\bar{n}]$ and (ii) f is concave on the direction $(\mathbf{e}_c)$ for any $c \in [\bar{n}]$, then $\mathbb{E}[f(\mathbf{x})] \leq f(\mathbf{x}^*)$.*

Proof of Theorem 2.7. Combining Lemma A.1 and Lemma A.2, we have the following inequalities:

$$\Pr[v(U) \leq t] \overset{(i)}{\leq} \mathbb{E}[g_{t,\theta}(\mathbf{x})] \overset{(ii)}{\leq} g_{t,\theta}(\mathbf{x}^*) \overset{(iii)}{\leq} \exp(-\delta^2 \mu/2).$$

The inequality (i) is due to (A.1a) of Lemma A.1, by taking the expectation over the distribution $\mathbf{x}$ produced by Algorithm 1 on both sides. The inequality (ii) is due to Lemma A.2, (A.1c) and (A.1d). The inequality (iii) is due to (A.1b) of Lemma A.1. Since $t = (1-\delta)\mu$, we have the desired concentration bound for items selected by Algorithm 1. □

We remark that similar to the standard Chernoff bound, Theorem 2.7 holds for cases where we only know a lower bound of the actual expected value.

Proof of Lemma A.2. The proof is similar to the pipage rounding proof. Fix an iteration in Algorithm 1. Let $\mathbf{x}^0$ be the vector $\mathbf{x}$ at the beginning of the iteration, and $\mathbf{x}^1$ be the vector at the end. If Algorithm 1 uses Operation 1 in this step, then $\mathbf{x}^1$ is $\mathbf{x}^0 + \delta_2 \mathbf{e}_a$ with probability $\frac{\delta_1}{\delta_1 + \delta_2}$, and $\mathbf{x}^0 - \delta_1 \mathbf{e}_a$ with probability $\frac{\delta_2}{\delta_1 + \delta_2}$. Since f is concave on the direction $\mathbf{e}_a$, we have:

$$\mathbb{E}[f(\mathbf{x}^1) \mid \mathbf{x}^0] = \frac{\delta_2}{\delta_1 + \delta_2} \cdot f(\mathbf{x}^0 - \delta_1 \mathbf{e}_a) + \frac{\delta_1}{\delta_1 + \delta_2} \cdot f(\mathbf{x}^0 + \delta_2 \mathbf{e}_a) \leq f\left(\mathbf{x}^0 + \frac{-\delta_2 \delta_1 + \delta_1 \delta_2}{\delta_1 + \delta_2} \cdot \mathbf{e}_a\right) = f(\mathbf{x}^0).$$

If Algorithm 1 uses Operation 2 in this step, then $\mathbf{x}^1$ is $\mathbf{x}^0 + \delta_2(\mathbf{e}_a - \mathbf{e}_b)$ with probability $\frac{\delta_1}{\delta_1 + \delta_2}$, and $\mathbf{x}^0 - \delta_1(\mathbf{e}_a - \mathbf{e}_b)$ with probability $\frac{\delta_2}{\delta_1 + \delta_2}$. Since $f(\cdot)$ is concave on the direction $(\mathbf{e}_a - \mathbf{e}_b)$, we have:

$$\mathbb{E}[f(\mathbf{x}^1) \mid \mathbf{x}^0] = \frac{\delta_2}{\delta_1 + \delta_2} \cdot f(\mathbf{x}^0 - \delta_1(\mathbf{e}_a - \mathbf{e}_b)) + \frac{\delta_1}{\delta_1 + \delta_2} \cdot f(\mathbf{x}^0 + \delta_2(\mathbf{e}_a - \mathbf{e}_b)) \leq f(\mathbf{x}^*).$$

We obtain $\mathbb{E}[f(\mathbf{x})] \leq f(\mathbf{x}^*)$ by induction over all iterations. This finishes the proof of Lemma A.2. $\square$

B   Missed Proofs from Section 3

B.1   Proof of Lemma 3.1

The following theorem considers the *submodular covering* problem:

Theorem B.1. *Suppose we are given an oracle to a monotone submodular function $v : 2^{[\bar{n}]} \to \mathbb{R}_{\geq 0}$, item costs $(c_i \in \mathbb{Q}_{>0})_{i \in [\bar{n}]}$, and a target value $V \geq 0$. Let $\mathrm{OPT} = \min_{S : v(S) \geq V} \sum_{i \in S} c_i$. Then for any constant $\epsilon > 0$, there is a polynomial time algorithm that finds a set S with $\sum_{i \in S} c_i \leq \mathrm{OPT}$ and $v(S) \geq \left(1 - \frac{1}{\mathrm{e}} - \epsilon\right) V$.*

The theorem can be obtained using the $(1 - \frac{1}{\mathrm{e}})$-approximation for the submodular maximization problem with a knapsack constraint [33] and binary search over OPT. We can scale c_i values so that they become integers. The binary search technique is needed to handle the case where c_i's are not polynomially bounded after scaling. In our algorithm, we can not afford to lose a multiplicative factor on the costs of items; therefore, we need this theorem.

Proof of Lemma 3.1. We shall use the ellipsoid method and utilize Theorem B.1 as a subroutine of the separation oracle. Similar to [11], we can first compute the maximum weighted NSW objective Φ that can be achieved by assigning one item to each agent, using the maximum-weight bipartite matching algorithm. Then, we know the value of the configuration LP is between $\ln \Phi$ and $\ln(m\Phi)$. By making $O(\frac{\log m}{\epsilon})$ guesses, we can assume we have a number o such that the value of (Conf-LP) is in $(o, o + \frac{\epsilon}{3}]$.

We now set up a dual program (Dual-LP) of (Conf-LP), with the objective replaced by a linear constraint.

(Dual-LP)

$$\sum_{j \in M} \alpha_j + \sum_{i \in N} \beta_i \leq o, \tag{5}$$

$$\sum_{j \in S} \alpha_j + \beta_i \geq w_i \cdot \ln v_i(S), \qquad \forall i \in N, S \subseteq M \tag{6}$$

$$\alpha_j \geq 0, \qquad \forall j \in M \tag{7}$$

As we have that the value of (Conf-LP) is in $(o, o + \epsilon]$, (Dual-LP) is infeasible. Given two vectors $\boldsymbol{\alpha} \in \mathbb{R}_{\geq 0}^M$ and $\boldsymbol{\beta} \in \mathbb{R}^n$ with $\sum_{j \in M} \alpha_j + \sum_{i \in N} \beta_i \leq o$, we know (6) is not satisfied for some $i \in N$ and $S \subseteq M$. Our goal is to design an approximate separation oracle for the inequality above.

By enumeration, we can guess the agent $i \in N$, and the item $j^* \in S$ with the maximum $v_i(j^*)$. We discard the items j with $v_i(j) > v_i(j^*)$. As we know $j^* \in S$, we know $v_i(S) \in [v_i(j^*), m v_i(j^*)]$.

We can then divide the interval into $O(\frac{\log m}{\epsilon})$ intervals of the form $[a, b]$ with $b \leq (1 + \epsilon)a$. Then we guess the interval $[a, b]$ where $v_i(S)$ resides in. Finally, we apply Theorem B.1, with item costs α_j and target value a. Then, we find a set $S' \ni j^*$ such that

$$\sum_{j \in S'} \alpha_j \leq \sum_{j \in S} \alpha_j \qquad \text{and} \qquad v_i(S') \geq \left(1 - \frac{1}{e} - \epsilon\right)a \geq \left(1 - \frac{1}{e} - \epsilon\right) \cdot \frac{v_i(S)}{1 + \epsilon}.$$

As (6) is infeasible for (i, S), we have

$$\sum_{j \in S'} \alpha_j + \beta_i \leq \sum_{j \in S} \alpha_j + \beta_i < w_i \cdot \ln v_i(S) \leq w_i \cdot \ln\left(\left(\frac{e}{e - 1} + O(\epsilon)\right) \cdot v_i(S')\right).$$

Using the same argument as in [11], we conclude that we can solve Equation (Conf-LP) up to an additive error of $\ln\left(\frac{e}{e-1} + O(\epsilon)\right) + \epsilon$. We can scale ϵ by a constant at the beginning so that the additive error term is at most $\ln\left(\frac{e}{e-1} + \epsilon\right)$. $\qquad \square$