Cycle Cancellation for Submodular Fractional Allocations and Applications

Chandra Chekuri
University of Illinois at Urbana Champaign, IL, USA

Pooja Kulkarni[1]
University of Chicago, IL, USA

Ruta Mehta
University of Illinois at Urbana Champaign, IL, USA

Jan Vondràk
Stanford University, CA, USA

Abstract

We consider discrete allocation problems where m indivisible goods need to be allocated among n agents. When agents' valuation functions are additive, the well-known cycle canceling lemma [39, 47, 45] plays a key role in the design and analysis of rounding algorithms. In this paper, we prove an analogous lemma for the case of submodular valuations. Our algorithm removes cycles in the support graph of a fractional allocation while guaranteeing that each agent's value, measured using the multilinear extension, does not decrease.

We demonstrate applications of the cycle-canceling algorithm, along with other ideas, to obtain new algorithms and results for three well-studied allocation objectives: max-min (Santa Claus problem), Nash social welfare (NSW), and maximin-share (MMS). For the submodular NSW problem, we obtain a $\frac{1}{5}$-approximation; for the MMS problem, we obtain a $\frac{1}{2}(1-1/e)$-approximation through new simple algorithms. For various special cases where the goods are "small" valued or the number of agents is constant, we obtain tight/best-known approximation algorithms. All our results are in the value-oracle model.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Submodular Optimization, Allocation Problems, Fair Division

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.21

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2511.21099

Funding *Chandra Chekuri*: Supported in part by NSF grant CCF-2402667.
Pooja Kulkarni: Supported in part by NSF grant CCF-2334461 and in part by a gift from Adobe to S.Khuller.
Ruta Mehta: Supported in part by NSF grant CCF-2334461.

Acknowledgements We would like to thank László Végh for useful discussions, in particular posing a question about using continuous local search for solving the multilinear relaxation of the Nash social welfare problem.

1 Introduction

Allocation problems are fundamental to both theory and practice. They arise naturally in diverse applications from optimization to fair division, and have been an influential source of techniques in discrete (combinatorial) optimization. In this paper we give a new tool for addressing allocation problems with submodular valuations.

[1] Most of this work was done when the author was a student at UIUC and a postdoc at Northwestern University.

Consider the problem of allocating a set $\mathcal{M}$ of m *indivisible* items to a set $\mathcal{N}$ of n agents with diverse preferences. An allocation can be viewed as a partition of $\mathcal{M}$ into sets $S_1, S_2, \ldots, S_n$ with S_i being the set given to agent i. Each agent i has a valuation function $v_i : 2^{\mathcal{M}} \to \mathbb{R}_+$ where $v_i(S)$ is the value that they obtain for subset $S \subseteq \mathcal{M}$. We will assume v_is to be *monotone submodular*[2], capturing decreasing marginal returns, and normalized so that $v_i(\emptyset) = 0$. Throughout this work, we denote an instance of the allocation problem by $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ and the set of all allocations as $\Pi_{|\mathcal{N}|}(\mathcal{M})$.

Allocation problems with submodular valuations are extensively studied within optimization [43, 20, 25] and fair division [42, 35, 48, 32, 29] communities under various objective functions. Also, see Section 1.1 for more detailed related works. A prominent example is to allocate items to maximize social welfare: find an allocation $S_1, \ldots, S_n$ to max $\sum_{i\in\mathcal{N}} v_i(S_i)$. While this is trivial for additive valuations (simply assign each good to the agent that values it the most), it is an interesting optimization problem for submodular valuations, and is NP-Hard. A simple greedy algorithm achieves an approximation ratio of 1/2 [44, 38] while a much more sophisticated approach achieves a tight approximation ratio of $(1 - 1/e)$ [49, 15]. The latter result was instrumental in developing the multilinear relaxation approach for submodular maximization problems. In this paper, we are concerned with more challenging objectives arising from fairness, efficiency, and other considerations, with the goal of designing unifying tools and techniques. Three such objectives are: (i) Max-min allocation (also referred to as the (Submodular) Santa Claus problem) [42], (ii) Nash Social Welfare (NSW) [36], and (iii) Maximin-share (MMS) [13]. These problems are well-known to be challenging even when agents' valuation functions are additive, i.e, $v_i(S) = \sum_{j\in S} v_i(\{j\})$ for all $i \in \mathcal{N}$. See Section 1.1 for more details about these problems.

Relax-and-round is a natural approach for most of these problems, where first a good *fractional allocation* is obtained and then rounded to an integral allocation while incurring some loss. The purpose of this paper is to develop a widely applicable technical tool that helps round a fractional allocation effectively under submodular valuations. We further highlight the utility of this tool by instantiating it on the above mentioned three objectives. Our tool is inspired by a corresponding result in the additive setting: A fractional allocation is an $m \cdot n$ dimensional vector $\mathbf{x}$ which we interpret as consisting of values $x_{i,j}$ for $i \in \mathcal{N}$ and $j \in \mathcal{M}$; $x_{i,j} \in [0, 1]$ is the fractional amount of item j that is assigned to agent i. We require that $\sum_i x_{i,j} = 1$ for each item j. We let $\mathbf{x}_i$ denote the m-dimensional vector consisting of the variables $x_{i,j}$, $j \in [m]$. Each fractional allocation $\mathbf{x}$ defines an allocation graph $G_{\mathbf{x}}$ which is a bipartite graph with agents on one side and items on the other side and an edge between i and j labeled with the number $x_{i,j}$ iff $x_{i,j} > 0$; in other words it is a representation of the support of $\mathbf{x}$. For a fractional allocation $\mathbf{x}$ the value of agent i in the additive setting, denoted by $v_i(\mathbf{x}_i)$ is naturally defined as $\sum_j v_i(\{j\})x_{i,j}$. Then the following rounding procedure is known.

▶ **Lemma 1** ([39, 47]). *Let* $\mathbf{x}$ *be a fractional allocation and* v_i *additive functions. Then there is another allocation* $\mathbf{y}$ *in the support of* $\mathbf{x}$ *such that (i) the allocation graph* $G_{\mathbf{y}}$ *has no cycles and (ii) for all agents* i, $v_i(\mathbf{y}_i) \geq v_i(\mathbf{x}_i)$. *Furthermore, the allocation* $\mathbf{y}$ *can be rounded to an integral allocation* $\mathcal{A} = (A_1, \ldots, A_n)$ *such that for all agents* i, $v_i(A_i) \geq V_i(\mathbf{y}_i) - \alpha_i$ *where* $\alpha_i = \max_{j:y_{i,j}>0} v_i(\{j\})$.

[2] A real-valued set function $f : 2^{\mathcal{M}} \to \mathbb{R}_+$ is submodular if $f(A + g) - f(A) \geq f(B + g) - f(B)$ for any $A \subseteq B$ and $g \in B \setminus A$; in other words the function exhibits diminishing marginal utility. Monotonicity means that $f(A) \leq f(B)$ for all $A \subseteq B$.

▶ **Corollary 2.** *Suppose* $\mathbf{x}$ *is a fractional allocation such that* $\max_{j:x_{i,j}>0} v_i(\{j\}) \leq \epsilon v_i(\mathbf{x}_i)$ *for each agent* i*. Then there is an integral allocation* $\mathbf{z}$ *such that* $v_i(\mathbf{z}_i) \geq (1-\epsilon)v_i(\mathbf{x}_i)$*.*

The corollary immediately allows one to find a good approximation in the so-called "small" items scenario. One can view the lemma and the corollary as interpolating between the fully fractional setting (which can be solved optimally via LP) and the discrete setting in which no single item is too important to an agent. The utility of the lemma can be understood by considering an alternative rounding strategy: randomly allocate each item independently to an agent based on fractional solution. The parameters obtained by such a rounding are much weaker – a logarithmic factor in the number of agents is typically lost since one has to rely on concentration inequalities followed by a union bound.

Here we generalize the preceding lemma to the submodular setting. In order to do this we need to work with a continuous extension of submodular functions. As mentioned above, for maximization problems the multilinear extension [14] has played a natural role.

▶ **Definition 3** (Multilinear Extension of f)**.**

$$F(x) \coloneqq \sum_{S \subseteq V} f(S) \prod_{j \in S} x_j \prod_{j \in V \setminus S} (1 - x_j) \tag{1}$$

It inherits some useful properties of submodularity and moreover, its value and partial derivatives can be evaluated for a given point $\mathbf{x}$ via random sampling.

Our first contribution is to formulate and prove the following theorem.

▶ **Theorem 4.** *Given an instance of the allocation problem* $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ *with monotone submodular valuations* v_i*, and a fractional allocation* $\mathbf{x}$ *of the goods to the agents, there is a fractional allocation* $\mathbf{x}'$ *such that (i) the allocation graph* $G_{\mathbf{x}'}$ *is acyclic, and (ii)* $V_i(\mathbf{x}') \geq V_i(\mathbf{x})$ *for all agents* $i \in \mathcal{N}$*; where* V_i *is the multilinear extension of* v_i*.*

We state the preceding theorem in a non-constructive way. A polynomial-time algorithm requires one to use approximations due to the sampling aspects of the multilinear extension. We defer formal details of this to a full version of the paper since they are somewhat standard. An efficient version of the lemma yields an allocation that loses a $(1 - o(1))$-factor in the value for each agent which we ignore in this version. See the end of Section 2 for more details.

Next we discuss application of this lemma to obtained improved algorithms and/or results for the three objectives mentioned above.

1.1 Applications

We demonstrate the utility of Theorem 4 on three problems that arise in fair allocation. The focus is not on obtaining improved approximation factors, but rather to illustrate how the cycle canceling procedure can be applied in conjunction with other ideas to design new relaxation-based algorithms. Table 1 gives a summary of our results.

All the three applications we discuss in this paper have seen a lot of work in the literature. While the table gives the current best-known guarantees, we discuss more details here.

[3] The results from [43] are for welfare maximization. However, the structure of instance used for the reduction directly implies the same bound for the problems mentioned here.

Table 1 Summary of Applications of Theorem 4. For the result marked with (*), while our result matches the existing result, we need ϵ to be a constant independent of n in this paper whereas the prior work required ϵ to be $\Omega(1/\log n)$. Results marked with (**) are information theoretic lower bounds whereas the other results are NP-hardness.

Fairness Notion	Problem Setting	Lower Bound	Prior Work	This Paper
Max-min	$v_i(j) \leq \epsilon\mathsf{OPT}\ \forall i,j$	–	–	$(1-1/e-\epsilon)$
Max-min	Constant n	$(1-1/e+\epsilon)$ [43] **	$(1-1/e-\epsilon)$ [20]	$(1-1/e-\epsilon)$*
NSW	General	$(1-1/e+\epsilon)$ [32]	$\approx 1/3.9$ [11]	1/5
NSW	Constant n	$(1-1/e+\epsilon)$ [32]	$(1-1/e-\epsilon)$ [32]	$(1-1/e-\epsilon)$*
MMS	General	$(1-1/e+\epsilon)$ [43][3]**	≈ 0.370 [48]	≈ 0.316
MMS	$v_i(j) \leq \epsilon\mathsf{MMS}_i\ \forall i,j$	–	≈ 0.370 [48]	$(1-1/e-\epsilon)$

1.1.1 Santa Claus (Max-min)

In Section 3.1, we investigate the max-min objective where given an instance of the fair allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$, the goal is to output an allocation $\mathcal{A} = \{A_1, \ldots, A_n\} \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ that maximizes $\min_{i\in\mathcal{N}} v_i(A_i)$. It is alternatively referred to as the Santa Claus problem.

1.1.1.1 Related work

Max-min was first considered as a fair allocation problem by [42]. There has been substantial work since then. Despite this, the approximability is still not well understood even in the additive setting. [12] showed that one cannot approximate max-min to a factor better than 2 in the additive setting unless P = NP. On the positive side, [8] gave an $O(\frac{\log\log m}{\log\log\log m})$-approximation algorithm in the restricted assignment setting (i.e., for each agent $v_i(\{j\}) \in \{v_j, 0\}$). The current best approximation for this setting is $\frac{1}{4+\epsilon}$ [22]. In the general setting, [17] gave a $\tilde{\Omega}(\frac{1}{n^\epsilon})$ approximation algorithm in time $n^{O(1/\epsilon)}$; in particular this implies a $\Omega(\frac{1}{poly(\log n)})$-approximation in quasi-polynomial time. These results were extended to the submodular setting: [7] gave a $\Omega(\frac{1}{poly(\log n)})$-approximation algorithm with running time $n^{O(\log n/\log\log n)}$, and for any fixed $\epsilon > 0$; a $\tilde{\Omega}(\frac{1}{n^\epsilon})$-approximation algorithm with running time $n^{O(1/\epsilon)}$. For the restricted case with submodular valuations, [6] gave an $\Omega(\frac{1}{\log\log n})$-approximation algorithm. Here, restricted submodular Santa Claus is defined as follows: there is a common submodular valuation function v, each agent has a subset $X_i \subseteq \mathcal{M}$ and the value of agent i is $v_i(S) = v(S \cap X_i)$. Another result known for additive valuations was by [12] who showed that there is always an allocation which gives each agent a value of at least $(\mathsf{OPT} - \max_{i\in\mathcal{N}, j\in\mathcal{M}} v_i(j))$ where OPT is the optimal max-min value, and this is via the cycle-cancellation lemma.

1.1.1.2 Our Contributions

In this work, we study the *submodular* Santa Claus problem. We show an additive approximation guarantee that is analogous to the result in [12] for additive valuations. While additive and multiplicative approximation guarantees are incomparable in general, when the goods are small valued, additive approximation imply multiplicative approximations. We thereby achieve the first non-trivial multiplicative approximation for the small-goods case with submodular valuations. Furthermore, we give a $(1-1/e-\epsilon)$ approximation for the case when the instance has a fixed number of agents. This was previously known from [20], however, we achieve a more efficient algorithm (see Remark 15).

1.1.2 Nash social welfare (NSW)

In Section 3.2, we investigate Nash social welfare which is a popular fairness *and* efficiency notion. The objective is to find an allocation $\mathcal{A} = (A_1, \ldots, A_n)$ that maximizes the geometric mean of agent valuations i.e., $\mathcal{A} = \text{argmax}_{S \in \Pi_{|\mathcal{N}|}(\mathcal{M})} \left(\prod_{i \in [n]} v_i(S_i)\right)^{1/n}$.

1.1.2.1 Related work

The Nash social welfare (NSW) objective has been proved to offer a natural balance between utilitarian welfare (sum of utilities) and fairness (max–min welfare) [36, 16]. Even for additive valuations, maximizing NSW is APX-hard [37] and obtaining a constant factor approximation was challenging. The first constant factor for additive valuations was presented by Cole and Gkatzelis [24], using a market equilibrium approach. Alternative approaches were found afterwards, using either market equilibria or fractional relaxations, that provide constant factors for additive, budget-additive and piecewise-linear concave valuations [4, 5, 23, 10, 28]. For submodular valuations, [32] gave the first approximation independent of the number of goods using a "matching-rematching" along with greedy allocation approach. [30] extended the matching-rematching approach along with fractional allocations and proved a constant factor for "Rado valuations" (a subclass of submodular valuations related to matroids). This framework was thereafter extended to a 1/380-approximation for submodular valuations by [40], which was subsequently improved to a 1/4-approximation using matching-rematching and local search [29]. A recent line of work [27, 26, 11] studied the *weighted* NSW problem, maximizing the weighted geometric mean of valuations, and obtained constant-factor approximations for additive and submodular valuations even in this setting. For additive valuations, their guarantees match the best known bound for unweighted NSW ($e^{1/e}$) from [10]; for submodular valuations, the most recent approximation factor is $\sim \frac{1}{3.56}$ [11] via computer-aided analysis. These algorithms are based on a configuration-type relaxation and randomized rounding.

1.1.2.2 Our Contributions

Our primary result here is to present a simple algorithm that still achieves a small constant-factor approximation for submodular NSW. Our algorithm is based on the matching-relaxation-rematching framework like [32, 30, 40] and provides a 1/5-approximation guarantee. Additionally, we also give a $(1 - 1/e - \epsilon)$-approximation for the case when the instance has constantly many agents in a more efficient manner than was previously done by [32] (see Remark 21).

1.1.3 Maximin Share (MMS)

In Section 3.3, we look at the Maximin share objective. An agent's MMS value, denoted by MMS_i is the maximum value she can ensure for herself if she divides the goods into n parts and chooses the worst part for herself.

1.1.3.1 Related work

Maximin Share (MMS) was defined as a fairness notion by [13]. [46] proved that MMS-allocations i.e., allocations that give every agent their MMS value need not exist even with additive valuations. Therefore, a series of works that obtain α-MMS allocation, i.e., each agent receives a value of α-MMS have been studied for additive valuations [46, 3, 33, 34, 2, 1]

culminating in a $3/4 + O(1)$-approximation. The algorithms in these works are mostly combinatorial and rely on either matching-based or greedy-based techniques. For general valuations, [9, 35] initiated the study for approximate MMS allocations. [9] gave a $\sim 1/10$ approximation algorithm via round-robin allocation and [35] gave a 1/3-approximation via local search. Recently, [48] gave an algorithm that provides 10/27-approximation to MMS using a greedy algorithm. For a subclass of submodular valuations called SPLC, [19] gave a 1/2 approximation algorithm based on market-equilibrium solution followed by rounding. Several other works on MMS on other valuation functions or in constrained settings exist that we do not list here.

1.1.3.2 Our Contributions

Our main theorem naturally extends the equilibrium-rounding approach to submodular valuations to give a $\frac{1}{2}(1-1/e-o(1))$-approximation algorithm. Additionally, when the goods are small compared to the MMS value (at most ϵMMS), we obtain a novel $(1-1/e-\epsilon)$-MMS allocation using the same approach.

2 Cycle-Cancellation under Multilinear Extension

In this section we present the main technical contribution of this work – Theorem 4. This theorem states that given an instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ of the allocation problem and any fractional allocation $\mathbf{x}$, we can convert it to an allocation with at most $|\mathcal{N}|-1$ fractional variables whose allocation graph forms a forest, while retaining the value received by any agent *under the multilinear relaxation*. It is well-known that when the function is submodular, the multilinear extension is convex in any $\mathbf{e}_i - \mathbf{e}_j$ direction [14]. We first generalize this to show that the multilinear extension is convex in any direction given by $t_i\mathbf{e}_i - t_j\mathbf{e}_j$, where $t_i, t_j \geq 0$. This allows us to argue how the function value changes when we move goods along a cycle allowing us to eventually cancel cycles in the allocation graph.

▶ **Theorem 4.** *Given an instance of the allocation problem $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with monotone submodular valuations v_i, and a fractional allocation $\mathbf{x}$ of the goods to the agents, there is a fractional allocation $\mathbf{x}'$ such that (i) the allocation graph $G_{\mathbf{x}'}$ is acyclic, and (ii) $V_i(\mathbf{x}') \geq V_i(\mathbf{x})$ for all agents $i \in \mathcal{N}$; where V_i is the multilinear extension of v_i.*

Before we prove the theorem, we see what happens to the value of the multilinear extension when for two goods i, j with $i \neq j$, we increase the allocation of good i by t_i and decrease the amount of good j by t_j. We remark that an equivalent lemma appears in an independent work on the knapsack problem with a submodular objective [41].

▶ **Lemma 5.** *Given a submodular function f with multilinear extension F, and allocations $\mathbf{x} = (x_1, \ldots, x_m)$ and $\bar{\mathbf{x}} = (x_1, \ldots, x_i + t_i, \ldots, x_j - t_j, \ldots, x_m)$, for any t_i, t_j such that either both $t_i \geq 0$ and $t_j \geq 0$, or both $t_i \leq 0$ and $t_j \leq 0$, we have,*

$$F(\bar{\mathbf{x}}) - F(\mathbf{x}) \geq t_i \frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}} - t_j \frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}}.$$

Proof. Let $\mathbf{x} = (x_1, \ldots, x_m)$, $\bar{\mathbf{x}} = (x_1, \ldots, x_i + t_i, \ldots, x_j - t_j, \ldots, x_m)$ and $P_{\mathbf{x}}(R) = \prod_{g\in R} x_g \prod_{g\in[m]\setminus R}(1 - x_g)$. Note that $\mathbf{x}$ and $\bar{\mathbf{x}}$ differ in probabilities only on the goods i and j. Therefore, when we expand the multilinear extension, the only places the terms differ are on the probabilities of selecting (or not selecting) i or j. We explicitly separate these terms. Towards this, we denote "$R : i, j$" to indicate all sets that include the elements i and j,

"$R : \tilde{i}, \tilde{j}$" are all sets that do not include the elements i and j. "$R : \tilde{i}, j$" and "$R : i, \tilde{j}$" are defined analogously. Further, we define $P_{\downarrow i,j}(R) = \prod_{g \in R \setminus \{i,j\}} x_g \prod_{g \in [m] \setminus R \setminus \{i,j\}} (1 - x_g)$ i.e., the probability of picking a set R according to a distribution $\mathbf{x}$ except that we ignore the probabilities corresponding to i and j. With this notation, we can write the difference of multilinear extensions as:

$$\begin{aligned}
F(\bar{\mathbf{x}}) - F(\mathbf{x}) &= \sum_R (P_{\bar{\mathbf{x}}}(R) - P_{\mathbf{x}}(R)) f(R) \\
&= \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)(t_j(1-x_i) - t_i(1-x_j) - t_i t_j) f(R) \\
&\quad + \sum_{R:i,\tilde{j}} P_{\downarrow i,j}(R)(t_i t_j + t_i(1-x_j) + x_i t_j) f(R) \\
&\quad + \sum_{R:\tilde{i},j} P_{\downarrow i,j}(R)(t_i t_j - (1-x_i)t_j - t_i x_j) f(R) \\
&\quad + \sum_{R:i,j} P_{\downarrow i,j}(R)(t_i x_j - x_i t_j - t_i t_j) f(R).
\end{aligned}$$

We rewrite the above equations by combining the terms in $t_i t_j$, t_i and t_j together and we get the following equation

$$\begin{aligned}
F(\bar{\mathbf{x}}) - F(\mathbf{x}) =& t_i t_j \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)(-f(R) + f(R+i) + f(R+j) - f(R+i+j)) \\
&+ t_i \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)(-(1-x_j)f(R) + (1-x_j)f(R+i) - x_j f(R+j) + x_j f(R+i+j)) \\
&+ t_j \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)((1-x_i)f(R) - x_i f(R+i) - x_j f(R+j) - x_i f(R+i+j))
\end{aligned}$$

Let $f(R+i) + f(R+j) - f(R) - f(R+i+j) = \alpha_R$. Note that the coefficient of t_i is $\frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}}$ and that of t_j is $-\frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}}$. Substituting these values, we get

$$F(\bar{\mathbf{x}}) - F(\mathbf{x}) = t_i t_j \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)\alpha_R + t_i \frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}} - t_j \frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}} \geq t_i \frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}} - t_j \frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}}$$

where the second inequality follows since $\alpha_R \geq 0$ for submodular valuation functions and we assumed t_i, t_j to be non-negative. ◀

▶ Remark 6. We note that it is well-known that when f is a submodular function, its multilinear extension F is convex along the direction $e_i - e_j$ for any $i, j \in [m]$. The above lemma proves a stronger version of this result i.e., the multilinear extension is convex along any $t_i e_i - t_j e_j$ for any $i, j \in [m]$ and $t_i, t_j \geq 0$.

We can now prove Theorem 4. Recall that the instance of allocation problem is denoted by $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ and we assume that each v_i is submodular. We denote the multilinear extension of v_i by V_i. Note that while we write $V_i(\mathbf{x})$, the value of V_i only depends on the part of $\mathbf{x}$ corresponding to agent i's allocation $\mathbf{x}_i$. Additionally, we use $V'_{i,j}$ to denote the partial derivative of V_i at $\mathbf{x}_i$ with respect to good j.

Proof of Theorem 4. The main idea is that if there is a cycle with fractional values x_{ij}, we can move values along the cycle consistently so that no agent loses in value, and some variable becomes an integer. For any given allocation $\mathbf{x}$, consider a cycle $a_0 - g_0 - a_1 - g_1 -$

$\cdots - a_{\ell-1} - g_{\ell-1} - a_0$ where a_i are the agents and g_i are the goods. To reallocate along the cycle, we need to maintain two constraints: (1) the total fractional allocation of each good remains constant, (2) no agent's value (under the multilinear extension) decreases. To maintain the first constraint, we will change the allocation of good i by δ_i on one edge and $-\delta_i$ on the other adjacent edge. Therefore, a reallocation strategy looks as follows:

$$x'_{a_i g_i} = x_{a_i g_i} + \delta_i$$
$$x'_{a_{i+1} g_i} = x_{a_{i+1} g_i} - \delta_i$$

with index operations modulo ℓ. By Lemma 5, for agent a_i not to lose value, it is sufficient to satisfy:

$$\delta_i V'_{a_i,g_i} - \delta_{i-1} V'_{a_i,g_{i-1}} \geq 0 \tag{2}$$

where, the index operations are modulo ℓ. We get ℓ such inequalities corresponding to the ℓ agents on the cycle. We claim that there is always a non-trivial solution to these inequalities, for any choice of V'_{a_i,g_i}, $V'_{a_i,g_{i-1}}$.

First, if $V'_{a_i,g_i} = 0$ for some agent i, we can set $\delta_{i'} = 0$ for all $i' \neq i$, and choose $\delta_i < 0$ to ensure that the gain of every agent is non-negative. Similarly, if $V'_{a_i,g_{i-1}} = 0$, we can set $\delta_{i'} = 0$ for all $i' \neq i-1$ and choose $\delta_{i-1} > 0$ to ensure that the gain of all agents is non-negative.

Hence, we can assume that $V'_{a_i,g_i} > 0$ and $V'_{a_i,g_{i-1}} > 0$ for all agents on the cycle. In this case, for any value δ_0, we can iteratively find values of $\delta_1, \ldots, \delta_{\ell-1}$ of the same sign such that $\delta_i V'_{a_i,g_i} - \delta_{i-1} V'_{a_i,g_{i-1}} = 0$ for $1 \leq i \leq \ell - 1$. These conditions mandate that $\delta_i = \delta_{i-1} \frac{V'_{a_i,g_{i-1}}}{V'_{a_i,g_i}}$, so we obtain inductively

$$\delta_{\ell-1} = \delta_0 \frac{V'_{a_1,g_0}}{V'_{a_1,g_1}} \frac{V'_{a_2,g_1}}{V'_{a_2,g_2}} \cdots \frac{V'_{a_{\ell-1},g_{\ell-2}}}{V'_{a_{\ell-1},g_{\ell-1}}}.$$

The last remaining condition is that

$$\delta_0 V'_{a_0,g_0} - \delta_{\ell-1} V'_{a_0,g_{\ell-1}} \geq 0$$

which is equivalent to

$$\delta_0 V'_{a_0,g_0} - \delta_0 \frac{V'_{a_1,g_0}}{V'_{a_1,g_1}} \frac{V'_{a_2,g_1}}{V'_{a_2,g_2}} \cdots \frac{V'_{a_{\ell-1},g_{\ell-2}}}{V'_{a_{\ell-1},g_{\ell-1}}} V'_{a_0,g_{\ell-1}} \geq 0.$$

We still have a free choice of $\delta_0 \neq 0$, so we can choose its sign appropriately so that the last condition is satisfied. By construction, this satisfies condition (2) for all agents i.

Denote by $\mathbf{x}'$ the modified fractional solution where $x'_{a_i,g_i} = x_{a_i,g_i} + \delta_i$. By the discussion above, all the δ_i's are determined by δ_0. We want to choose the magnitude of δ_0 so that $\mathbf{x}'$ is feasible and at least one variable on the cycle becomes an integer. This can be accomplished by choosing the supremum or infimum of δ_0 (depending on the desired sign) such that $\mathbf{x}'$ defined by the changes described above is still feasible. We claim that such $\mathbf{x}'$ must have a new integer variable: If all the variables on the cycle are fractional, then there is a larger/smaller value of δ_0 that we can choose and $\mathbf{x}'$ is still feasible. An extreme choice of δ_0 still defines a feasible solution, because the assignment polytope is a closed set.

Finally, we argue that the value for any agent cannot decrease in this operation. After choosing δ_0 as above, the remaining δ_i for $i \in [\ell]$ are chosen as $\delta_i = \delta_{i-1} \frac{V'_{a_i,g_{i-1}}}{V'_{a_i,g_i}}$. Since the partial derivatives are all positive, all δ_is will have the same sign. Therefore, by Lemma 5, we have $V_{a_i}(\mathbf{x}') \geq V_{a_i}(\mathbf{x})$. Note that it is in this step that we use the convexity of the multilinear extension along the direction that we modify our fractional solution. ◀

Note that the proof does not work for non-monotone submodular functions. We could possibly define δ_i to satisfy (2) even for non-monotone submodular functions, but since two variables could be increasing or decreasing at the same time, the relevant multilinear function could be concave rather than convex along the direction of change.

Notes on computational aspects of the procedure: In the discussion above, we ignore the computational issues of computing or estimating the values of $V_i(\mathbf{x})$ and $\frac{\partial V_i}{\partial x_{ij}}$. These issues can be handled by techniques similar to prior work using the multilinear extension: The values of $V_i(\mathbf{x})$ and its partial derivatives are expectations over a certain product distribution and hence can be estimated by random sampling. The sampling error can be made smaller than $\frac{\alpha_i}{poly(m,n)}$ where $\alpha_i = \max_{j \in \mathcal{M}: x_{ij} > 0} v_i(\{j\})$. Since variables are changing by at most 1 in each step, the true change in $V_i(\mathbf{x})$ might be off by $\frac{\alpha_i}{poly(m,n)}$ compared to our estimates. The number of rounding steps is polynomial in m and n (at least one new variable becomes an integer in each step), and hence we can choose the sampling parameters so that the loss for each agent is less than $\frac{\alpha_i}{poly(m,n)}$ at the end. In many applications, this additive error is negligible compared to the loss of α_i which we shall incur in any case due to the rounding on the remaining forest. For completeness, we give our computational result:

▶ **Theorem 7.** *Given an instance of the allocation problem $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ with monotone submodular valuations v_i, and a fractional allocation $\mathbf{x}$ of the goods to the agents, in randomized polynomial time, we can compute with high probability another fractional allocation $\mathbf{y}$ such that the support of $\mathbf{y}$ is acyclic, and $V_i(\mathbf{y}) \geq V_i(\mathbf{x}) - \frac{\alpha_i}{poly(m,n)}$ for all agents $i \in [n]$ where V_i is the multilinear extension of v_i and $\alpha_i = \max_{j \in \mathcal{M}: x_{ij} > 0} v_i(\{j\})$.*

3 Applications

In this section we discuss applications of the cycle-cancellation. All applications we discuss use the same rounding based on cycle-cancellation which we call NonUniformPipageRounding and is outlined in Algorithm 1. Similar rounding has appeared in the context where valuations are additive or SPLC[4] (eg., [24, 19]).

Algorithm 1 NonUniformPipageRounding$((\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}}), \mathbf{x})$.

1. Create $\mathbf{x}'$ from $\mathbf{x}$ using Theorem 4.
2. In the forest of $G_{\mathbf{x}'}$, root each tree at an arbitrary agent
3. Allocate each fractional good to its parent in the rooted tree

We can prove the following guarantee of this algorithm.

▶ **Lemma 8.** *Given an instance of the allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ and any fractional allocation $\mathbf{x}$, Algorithm 1 (NonUniformPipageRounding) finds an allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|}) \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ such that $v_i(A_i) \geq V_i(\mathbf{x}) - v_i(\ell_i)$ for all $i \in \mathcal{N}$, where $\ell_i = max_{j \in \mathcal{M}} v_i(\{j\})$.*

Proof. From Theorem 4, the agents lose no value in converting to $\mathbf{x}'$. Then, since the graph is acyclic, step 2 can be executed. Since each agent in the rounding can lost at most one good (the parent good), the agents lose a single good. By submodularity (in fact, subadditivity), the lemma follows. ◀

[4] SPLC is a class of functions that capture additive valuations and are a subset of submodular valuations.

▶ Remark 9. As mentioned at the end of previous section, we ignore the computational aspects of cycle-cancellation procedure. All our applications use the NonUniformPipageRounding where the rounding incurs loss of one good. The loss due to computational aspects of cycle-cancellation (result of Theorem 7) is negligible compared to this loss and therefore can be safely ignored. For keeping the notation clean, we do not add that loss here. Similar to the computational version of Theorem 4, we get the following computational version:

▶ **Lemma 10.** *Given an instance of the allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ and any fractional allocation $\mathbf{x}$, Algorithm 1* (NonUniformPipageRounding)*, in randomized polynomial time finds with high probability an allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|}) \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ such that $v_i(A_i) \geq V_i(\mathbf{x}) - v_i(\ell_i) - \frac{\ell_i}{poly(|\mathcal{N}|,|\mathcal{M}|)}$ for all $i \in \mathcal{N}$, where $\ell_i = max_{j\in\mathcal{M}} v_i(\{j\})$.*

3.1 Approximating Submodular Max-min

The max-min problem also known as the Santa Claus problem is defined as: given an instance of the allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$, output an allocation $\mathcal{A} = (A_i, \ldots, A_{|\mathcal{N}|}) \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ to maximize $\min_{i\in\mathcal{N}} v_i(A_i)$. This value is called the max-min value and we denote it as $\mathsf{OPT} = \max_{\mathcal{A}\in\Pi_{|\mathcal{N}|}(\mathcal{M})} \min_{i\in\mathcal{N}} v_i(A_i)$.

As discussed in Section 1.1.1, our primary aim is to provide an additive approximation for the Submodular Santa Claus problem. We show that this directly gives a multiplicative approximation for "small" goods. Additionally, we also show how to use cycle-cancellation to achieve a multiplicative approximation for the special case when the instance has a fixed number of agents. The key to these results is the NonUniformPipageRounding subroutine (Algorithm 1). To use the subroutine, we naturally need a fractional solution. To obtain this fractional solution for the max-min problem, we use the following result by [20, 18].

▶ **Theorem 11** ([20, 18])**.** *Let $f_1, \ldots, f_n$ be monotone non-negative submodular functions over a common ground set N, $B_1, B_2, \ldots, B_n \in \mathbb{R}_+$ and let $Q \in [0,1]^{|N|}$ be a solvable*[5] *polytope. For every fixed $\epsilon > 0$ there is a polynomial time randomized algorithm that either outputs a point $\mathbf{x} \in Q$ such that for all i,*

$$F_i(\mathbf{x}) \geq (1 - 1/e - \epsilon)B_i$$

or correctly decides that there is no point $\mathbf{x}$ in Q such that $F_i(\mathbf{x}) \geq B_i$ for all i.

▶ **Theorem 12.** *Consider an instance, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ of the Submodular Santa Claus problem with the optimal value OPT. Denote the maximum value of any good as $\mathrm{Max} = \max_{i\in\mathcal{N}, j\in\mathcal{N}} v_i(j)$. For any fixed $\epsilon > 0$, one can obtain in randomized polynomial time an allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|})$ such that $v_i(A_i) \geq (1 - 1/e - \epsilon)\mathsf{OPT} - \mathrm{Max}$.*

Proof. First, we obtain a fractional allocation $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_{|\mathcal{N}|})$ such that $V_i(\mathbf{x}_i) \geq (1 - 1/e - \epsilon)\mathsf{OPT}$. This can be done as follows. We guess the value of OPT via binary search. We then define Q as a partition matroid constructed as follows: for each good, create $|\mathcal{N}|$ copies, one corresponding to each agent. All copies of a given good belong to the same part, and there are $|\mathcal{M}|$ such parts, corresponding to each good. An independent set of this matroid can include at most one element from each part. This construction naturally defines an allocation: each selected copy corresponds to a specific agent, and that agent receives

[5] A polytope is called solvable if we can optimize linear functions over the polytope in polynomial time.

the associated good. The agents' valuation functions extend naturally while preserving submodularity: given an independent set of the matroid, the value F_i for agent i is computed by considering only the copies associated with that agent. This construction first appeared in [38]. Assuming that OPT is a valid estimate of a value that each agent can achieve, we use Theorem 11 to find an $\mathbf{x} \in Q$ that achieves $V_i(\mathbf{x}_i) \geq (1 - 1/e - \epsilon)OPT$ for each agent. To round this fractional solution, we apply the NonUniformPipageRounding to $\mathbf{x}$, and obtain an integral allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|})$ such that $v_i(A_i) \geq V_i(\mathbf{x}_i) - v_i(\ell_i)$ where ℓ_i is the largest valued good for agent i. Therefore, we obtain $v_i(A_i) \geq (1 - 1/e - \epsilon)\mathsf{OPT} - \text{Max}$. ◀

As a corollary, we obtain the following result when $\text{Max} \leq \epsilon\mathsf{OPT}$.

▶ **Corollary 13.** *Consider an instance of Submodular Santa Claus with a promise that there is a feasible allocation $(S_1^*, S_2^*, \ldots, S_{\mathcal{N}}^*)$ of value OPT such that $v_i(j) \leq \epsilon\mathsf{OPT}$ for each $i \in \mathcal{N}$ and each $j \in S_i$. In this setting, for any fixed $\epsilon' > 0$, there is an efficient randomized algorithm that with high probability either outputs an allocation $(S_1, S_2, \ldots, S_{|\mathcal{N}|})$ such that $v_i(S_i) \geq (1 - 1/e - \epsilon - \epsilon')\mathsf{OPT}$ or correctly refutes the promise.*

3.1.1 Constant Number of Agents

Here we consider the setting with a fixed number of agents. In the additive setting there is a PTAS for this case. For the submodular setting the following result can be obtained.

▶ **Theorem 14.** *For any fixed $\epsilon > 0$ and any fixed number of agents $|\mathcal{N}| \geq 1$, there is a $(1 - 1/e - \epsilon)$-approximation for the Submodular Santa Claus problem.*

Proof Sketch. We sketch the ideas since the essence of this has already been captured in Theorem 12. Suppose we have guessed OPT. Since $|\mathcal{N}|$ is fixed we can guess all the "large" items for each agent i from some fixed optimum allocation $(S_1^*, \ldots, S_{|\mathcal{N}|}^*)$. More formally, for each agent i we guess a set $A_i \subset S_i^*$ such that for all $j \in S_i^* \setminus A_i$, $v_i(A_i + j) - v_i(A_i) \leq \epsilon\mathsf{OPT}$. It is easy to see that there exists such a set with $|A_i| \leq (1/\epsilon + 1)$. Thus there are $|\mathcal{M}|^{O(|\mathcal{N}|/\epsilon)}$ guesses that the algorithm needs to try and this is polynomial when $|\mathcal{N}|$ is fixed. Once these are guessed we can then find a fractional allocation and round as in the proof of Theorem 12. The one good loss for an agent i in this setting is at most $\epsilon v_i(A_i)$ due to the preprocessing. The running time is dominated by the guessing step which incurs an overhead of $|\mathcal{M}|^{O(|\mathcal{N}|/\epsilon)}$. ◀

▶ Remark 15. One can use a different rounding approach for the second step after the guessing the large items and solving the relaxation. This was the approach that was outlined in [20] and is based on randomly allocating the items according the fractional allocation; more precisely we allocate each item j independently to exactly one agent where the probablity of agent i receiving item j is precisely $x_{i,j}$. Since the fractional items are "small", one can then use concentration properties of submodular functions, to argue that each agent will have their value preserved to within a $(1 - \epsilon)$-factor of their fractional value with sufficiently high probability; one then uses union bound over all agents. However, randomized rounding incurs larger errors than our rounding method. To make randomized rounding work, we need to use the fact that the number of agents is small, and we need $\epsilon = \Omega(1/\log|\mathcal{N}|)$ for the union bound rather than a fixed constant as above.

3.2 Nash social welfare with submodular valuations

Nash Social Welfare (NSW) is another possible objective in allocation problems: Given a set $\mathcal{M}$ of m items, and a set $\mathcal{N}$ of n agents with valuations $(v_i : i \in \mathcal{N})$, find an allocation $\mathcal{A} = (A_1, \ldots, A_n)$ to maximize the Nash social welfare objective,

$$\left(\prod_{i \in \mathcal{N}} v_i(A_i)\right)^{1/n}.$$

In this section, we present a new $\frac{1}{5}$-approximation algorithm for this problem with monotone submodular valuations, using the rounding lemma.

As mentioned in Section 1.1.2, our algorithm follows the matching-rematching approach that has been used for maximizing NSW with submodular valuations [32, 40, 29]. [40] used this framework with a log-multilinear relaxation and randomized rounding, which gave the first constant factor for NSW with submodular valuation [40]; however, the factor was rather small ($\approx 1/380$). Later, the factor was significantly improved to $1/4$ [29], by replacing the log-multilinear relaxation by discrete local search. Here we return to the log-multilinear relaxation and show that it can be actually used to extract a good approximation factor as well. This is accomplished primarily by the new rounding lemma. In addition, we provide an alternative way to solve the log-multilinear relaxation (approximately) by continuous local search; this also helps in improving the approximation factor, but the effect of this improvement is relatively minor (essentially replacing $1/e$ by $1/2$).

3.2.1 A $\frac{1}{5}$-Approximation Algorithm

Let us describe our new algorithm. It consists of 3 phases: (1) initial matching, (2) relaxation, (3) rematching. The relaxation phase is the new contribution here: we solve the log-multilinear relaxation using continuous local search, and then round it using the non-uniform rounding lemma. A formal description of the algorithm follows.

Algorithm 2 1/5-approximation for NSW with submodular valuations.

Input : Instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$
Output : An allocation of items with NSW at least $\frac{1}{5}$OPT

$\tau :=$ matching $\mathcal{N} \to \mathcal{M}$ maximizing $\sum_{i \in \mathcal{N}} \log v_i(\tau(i))$
$H := \tau[\mathcal{N}], \mathcal{M}' = \mathcal{M} \setminus H$
$\mathbf{x} :=$ ContinuousLocalSearch$(\mathcal{M}', \{v_i : i \in \mathcal{N}\})$
$(S_1, \ldots, S_n) :=$ NonUniformPipageRound$(\mathbf{x}, \{v_i : i \in \mathcal{N}\})$
$\sigma :=$ matching $\mathcal{N} \to \mathcal{M}$ maximizing $\sum_{i \in \mathcal{N}} \log v_i(S_i + \sigma(i))$
return $(S_i + \sigma(i) : i \in \mathcal{N})$

First, we have the following simple lemma about the initial matching.

▶ **Lemma 16.** *If $\tau : \mathcal{N} \to \mathcal{M}$ is the initial optimal matching, then for every agent $i \in \mathcal{N}$ and every remaining item $j \in \mathcal{M} \setminus \tau[N]$, $v_i(j) \le v_i(\tau(i))$.*

Proof. If $v_i(j) > v_i(\tau(i))$, then we could improve the matching by redefining $\tau(i) := j$; note that this item is not allocated to anybody else in the matching. ◀

Algorithm 3 ContinuousLocalSearch$(\mathcal{M}', (v_i : i \in \mathcal{N})$.

Let $P := \{\mathbf{x} \in \mathcal{R}_+^{\mathcal{M}' \times \mathcal{N}} : \forall j \in \mathcal{M}, \sum_{i \in \mathcal{N}} x_{ij} = 1\}$ (the assignment polytope)
Initialize $\mathbf{x}^{(0)} := \frac{1}{n}\mathbf{1} \in P$, and $t := 0$
Define the function $F(\mathbf{x}) = \sum_{i \in [n]} \log V_i(\mathbf{x}_i)$.
while $\exists \mathbf{y} \in P, (\mathbf{y} - \mathbf{x}) \cdot \nabla F(\mathbf{x}) > 0$ **do**
  $\mathbf{x}^{(t+1)} := \mathbf{x}^{(t)} + \epsilon(\mathbf{y} - \mathbf{x}^{(t)})$
  $t = t + 1$

Next, we analyze the continuous local search algorithm. Here we ignore numerical issues of convergence of the algorithm; we assume that the output is a local optimum, which can be implemented up to an arbitrarily small error. Note that here we do not state the computational versions of our results. To implement Algorithm 3 in polynomial time, we need to discretize the process like [21]. Let P denote the assignment polytope as in the description of ContinuousLocalSearch.

▶ **Lemma 17.** *Suppose that* $\mathbf{x} \in P$ *is the output of* ContinuousLocalSearch*, and* $\mathbf{x}^* \in P$ *is any other feasible solution. Then*

$$\sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i^*)}{V_i(\mathbf{x}_i)} \leq 2|\mathcal{N}|.$$

Proof. If $\mathbf{x}$ is a local optimum, then we have

$$(\mathbf{x}^* - \mathbf{x}) \cdot \nabla F(\mathbf{x}) = \sum_{i \in \mathcal{N}} (\mathbf{x}_i^* - \mathbf{x}_i) \cdot \nabla(\log V_i(\mathbf{x}_i)) \leq 0.$$

Consider the points $\mathbf{x} \vee \mathbf{x}^*$ and $\mathbf{x} \wedge \mathbf{x}^*$ (the coordinate maximum and minimum). We can write $(\mathbf{x} \vee \mathbf{x}^*) + (\mathbf{x} \wedge \mathbf{x}^*) = \mathbf{x} + \mathbf{x}^*$, and hence

$$\sum_{i \in \mathcal{N}} ((\mathbf{x}_i \vee \mathbf{x}_i^*) - \mathbf{x}_i) \cdot \nabla(\log V_i(\mathbf{x}_i)) \leq \sum_{i \in \mathcal{N}} (\mathbf{x}_i - (\mathbf{x}_i \wedge \mathbf{x}_i^*)) \cdot \nabla(\log V_i(\mathbf{x}_i)).$$

By the chain rule, $\nabla(\log V_i(\mathbf{x}_i)) = \frac{\nabla V_i(\mathbf{x}_i)}{V_i(\mathbf{x}_i)}$. Also, by submodularity of V_i, we have $((\mathbf{x}_i \vee \mathbf{x}_i^*) - \mathbf{x}_i) \cdot \nabla V_i(\mathbf{x}_i) \geq V_i(\mathbf{x}_i \vee \mathbf{x}_i^*) - V_i(\mathbf{x}_i)$. Similarly, $(\mathbf{x}_i - (\mathbf{x}_i \wedge \mathbf{x}_i^*)) \cdot \nabla V_i(\mathbf{x}_i) \leq V_i(\mathbf{x}_i) - V_i(\mathbf{x}_i \wedge \mathbf{x}_i^*)$ From here and the inequality above,

$$\sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i \vee \mathbf{x}_i^*) - V_i(\mathbf{x}_i)}{V_i(\mathbf{x}_i)} \leq \sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i) - V_i(\mathbf{x}_i \wedge \mathbf{x}_i^*)}{V_i(\mathbf{x}_i)}.$$

This inequality is valid even for non-monotone submodular functions. Since here we are dealing with monotone submodular functions, we can drop $V_i(\mathbf{x}_i \wedge \mathbf{x}_i^*)$ and estimate $V_i(\mathbf{x}_i \vee \mathbf{x}_i^*) \geq V_i(\mathbf{x}_i^*)$, which gives

$$\frac{1}{n} \sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i^*)}{V_i(\mathbf{x}_i)} \leq 2.$$ ◀

Next, we apply the non-uniform pipage rounding procedure. Given a fractional solution $\mathbf{x}$, it produces an allocation $(S_1, \ldots, S_n)$ of the items in $\mathcal{M}'$ such that

$$v_i(S_i) \geq V_i(\mathbf{x}_i) - v_i(\ell(i))$$

where

$$\ell(i) = \operatorname*{argmax}_{j \in \mathcal{M}'} v_i(j).$$

For the analysis of the last stage, we need the following "rematching lemma", which has appeared in various forms in this line of work. The variant we use here appears in [31].

▶ **Lemma 18.** *If $\tau : \mathcal{N} \to \mathcal{M}$ is an optimal initial matching, $H = \tau[\mathcal{N}]$, $\pi : \mathcal{N} \to H$ is any other matching on H, $(S_i : i \in \mathcal{N})$ is any partition of $\mathcal{M}' = \mathcal{M} \setminus H$, and $\ell(i) = \operatorname{argmax}_{j \in \mathcal{M}'} v_i(j)$, then there is a matching $\sigma : \mathcal{N} \to H$ such that*

$$\prod_{i \in \mathcal{N}} v_i(S_i + \sigma(i)) \geq \prod_{i \in \mathcal{N}} \max\{v_i(S_i), v_i(\ell(i)), v_i(\pi(i))\}.$$

Using the rematching lemma, we conclude the analysis as follows.

▶ **Theorem 19.** *If* OPT *is the* NSW *value of the optimal solution, then our algorithm finds a solution of value*

$$\left(\prod_{i \in \mathcal{N}} v_i(S_i + \sigma(i))\right)^{1/n} \geq \frac{1}{5}\mathsf{OPT}.$$

Proof. Let the optimal allocation be $(H_i \cup O_i : i \in \mathcal{N})$, where $H_i \subset H$ and $O_i \subset \mathcal{M}' = \mathcal{M} \setminus H$. Let $\pi : \mathcal{N} \to H$ be a matching such that $\pi(i) = \operatorname{argmax}_{j \in H_i} v_i(j)$ if $H_i \neq \emptyset$, and otherwise $\pi(i)$ an arbitrary item in H so that π is matching. By submodularity, we have

$$v_i(H_i \cup O_i) \leq v_i(O_i) + |H_i| v_i(\pi(i)) = V_i(\mathbf{x}_i^*) + |H_i| v_i(\pi(i))$$

where $\mathbf{x}_i^* = \mathbf{1}_{O_i}$.

From Lemma 18, we have that the value obtained by our algorithm is

$$\mathsf{ALG} = \left(\prod_{i \in \mathcal{N}} v_i(S_i + \sigma(i))\right)^{1/n} \geq \left(\prod_{i \in \mathcal{N}} W_i\right)^{1/n}$$

where $W_i = \max\{v_i(S_i), v_i(\ell(i)), v_i(\pi(i))\}$. From the pipage rounding procedure, we get $v_i(S_i) \geq V_i(\mathbf{x}_i) - v_i(\ell(i))$. Hence,

$$W_i \geq \max\{V_i(\mathbf{x}_i) - v_i(\ell(i)), v_i(\ell(i)), v_i(\pi(i)\} \geq \max\{\frac{1}{2}V_i(\mathbf{x}_i), v_i(\pi(i))\}.$$

Finally, we estimate by AM-GM

$$\begin{aligned} \frac{\mathsf{OPT}}{\mathsf{ALG}} &\leq \left(\prod_{i \in \mathcal{N}} \frac{v_i(H_i \cup O_i)}{W_i}\right)^{1/n} \leq \frac{1}{n}\sum_{i \in \mathcal{N}} \frac{v_i(H_i \cup O_i)}{W_i} \\ &\leq \frac{1}{n}\sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i^*) + |H_i| v_i(\pi(i))}{W_i} \leq \frac{1}{n}\sum_{i \in \mathcal{N}} \left(\frac{V_i(\mathbf{x}_i^*)}{\frac{1}{2}V_i(\mathbf{x}_i)} + |H_i|\right). \end{aligned}$$

Using Lemma 17 and the fact that $\sum_{i \in \mathcal{N}} |H_i| \leq n$, we conclude that $\frac{\mathsf{OPT}}{\mathsf{ALG}} \leq 5$. ◀

3.2.2 Constant number of agents

We can prove the following theorem for NSW with a constant number of agents. This theorem also works for weighted NSW. We note that a similar result was known previously [32]. See Remark 21 for a comparison.

▶ **Theorem 20.** *Given an instance of the fair allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$, we can compute in randomized polynomial time an allocation $A_i \in \Pi_n([m])$ such that the value of* NSW *of the allocation is $(1 - 1/e - \epsilon)$*OPT *where* OPT *is the value of the optimal Nash social welfare.*

Proof Sketch. The proof is similar to proof of constantly many agents for the max-min problem (see 14) and we only sketch the idea here. First, we predict the value agent i receives in an optimal allocation, say OPT_i. For a particular agent, we can guess this to a $(1+\epsilon)$ multiplicative approximation using a grid search in polynomial time. Therefore, for constantly many agents, we can guess the valuation profile for an optimal NSW solution in polynomial time. Now, consider the corresponding optimal allocation, $(S_1^*, \ldots, S_n^*)$. From each S_i^*, we guess a subset $A_i \subseteq S_i^*$ such that for each $g \in S_i^* \setminus A_i$, $v_i(g \mid A_i) \leq \epsilon\mathsf{OPT}_i$. It is clear that we can have an A_i such that $|A_i| \leq (1/\epsilon)$ otherwise, we have a value of more than OPT_i for agent i. Therefore, together for all n agents, this A_i can be guessed in $m^{O(n/\epsilon)}$ time. This is clearly polynomial when n and ϵ are constants. Now, we find a fractional solution for the NSW problem – for each agent, we define it's new valuation function to be $v_{i|A_i}$ which is the function that gives marginal values of sets on A_i. It is well-known and easy to see that this function is also submodular. For this function, we aim to allocate agent i value $\mathsf{OPT}_i - v_i(A_i)$. Again, following the Theorem 11 from previous section, if such an allocation exists, we can compute a fractional allocation $\mathbf{x}$ in time polynomial in m, n and ϵ' such that $v_i(\mathbf{x}_i) \geq (1-1/e-\epsilon')(\mathsf{OPT}_i - v_i(A_i))$. Further, we can also ensure that no agent is allocated a good of value more than $\epsilon \cdot \mathsf{OPT}$ by making the value of any such good for the agent zero (this also maintains submodularity). Then we run the non-uniform pipage rounding algorithm and obtain an allocation that allocates each agent a value of $(1-1/e-\epsilon'-\epsilon)\mathsf{OPT}_i$. Therefore, in time polynomial in m, n and $\epsilon'' = \epsilon' + \epsilon$, we can compute an allocation that maximizes NSW with constant number of agent up to a factor of $(1-1/e-\epsilon'')$ i.e., we have a PTAS for this problem with constantly many agents. Note that this algorithm also works with asymmetric agents. ◀

▶ Remark 21. A $(1-1/e-\epsilon)$-approximation for constant n was achieved previously [32]. That work relied on an integral version of Theorem 11 which in turn was based on the same ideas as in max-min allocation from [20]. Therefore, to obtain an approximation guarantee of $(1-1/e-\epsilon)$ for any fixed $\epsilon > 0$, [32] requires guessing from an optimal allocation all goods allocated to an agent i of value at least $\epsilon \cdot \mathsf{OPT}_i/\log n$. In the preceding theorem, for a fixed $\epsilon > 0$, we need to guess the "large" goods of value at least $\epsilon\mathsf{OPT}_i$. Therefore the size of the set to be guessed is reduced.

3.3 Approximating MMS for Submodular Valuations

MMS is another fairness objective frequently studied in allocation problems. It is a share-based guarantee, i.e., each agent demands a certain value a.k.a. share. An allocation is considered to be fair if all agents receive their share. In case of MMS, this value for a particular agent i depends only on the valuation function of the agent, v_i and the number of agents $|\mathcal{N}|$. It is defined as follows:

$$\mathsf{MMS}_i^{|\mathcal{N}|}(\mathcal{M}) := \max_{A \in \Pi_{|\mathcal{N}|}(\mathcal{M})} \min_{i \in [n]} v_i(A_i).$$

Essentially, the MMS value of an agent is the maximum value she can ensure for herself when she divides the goods into $|\mathcal{N}|$ parts and chooses the worst part for herself. When $\mathcal{M}$ and $\mathcal{N}$ are clear from the context, we denote $\mathsf{MMS}_i^{|\mathcal{N}|}(\mathcal{M})$ as simply MMS_i. As mentioned in Section 1.1.3, since MMS-allocations do not exist, we study α-MMS allocations for $\alpha(\leq 1)$ as large as possible. Here, we give an algorithm that achieves a value of $\approx 1/2(1-1/e) \approx 0.316$ based on a simple application of the NonUniformPipageRounding. Our approach is again to round a fractional allocation. While in the previous two sections we solved some fractional

program to find the fractional allocation, in this section, we give a direct solution using two known results. To state the results, we need to define another extension of submodular valuation functions called the concave extension. Given a submodular function, f the concave extension, f^+ of f is defined as follows.

▶ **Definition 22** (Concave extension of f)**.**

$$f^+(\mathbf{x}) = \max_{(\alpha_S)_{S\in 2^{\mathcal{M}}}} \{\sum_S f(S)\alpha_S : \sum_S \alpha_S = 1 \text{ and } \sum_{S\ni j} \alpha_S = x_j \quad \forall j \in \mathcal{M} \text{ and } \alpha_S \geq 0 \quad \forall S \in 2^{\mathcal{M}}\}.$$

Now, we state the aforementioned results. The first one, from [19] gives an upper bound on MMS in terms of the concave extension. The second one, from [14], is called the correlation gap and connects the value of the concave extension at any fractional point with value of the multilinear extension at the same point.

▶ **Lemma 23** ([19])**.** *Given an instance of the fair division problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with submodular valuations, $\mathsf{MMS}_i \leq v_i^+(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|}))$ for all $i \in \mathcal{N}$, where v_i^+ is the concave extension of the function v_i.*

▷ Claim 24 ([14]). For any monotone submodular function, f with concave extension f^+ and multilinear extension F,

$$F(\mathbf{x}) \geq \left(1 - \frac{1}{e}\right) f^+(\mathbf{x}).$$

Combining the above two results, we get,

▷ Claim 25. Given an instance of the fair division problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with submodular valuations, $\mathsf{MMS}_i \leq \left(\frac{e}{e-1}\right) \cdot V_i(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|}))$.

We can now use NonUniformPipageRounding to obtain:

▶ **Theorem 26.** *Given a fair division instance, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with submodular valuations, if $\max_{i,j} v_i(\{j\}) \leq \epsilon \cdot \mathsf{MMS}_i$ for all $i \in \mathcal{N}, j \in \mathcal{M}$, we can compute an allocation in randomized polynomial time where each agent receives a value of $\left(1 - \frac{1}{e} - \epsilon\right) \mathsf{MMS}_i$.*

Proof. We run NonUniformPipageRounding on the allocation $\mathbf{x} = (\frac{1}{|\mathcal{N}|}\mathbf{1}^{|\mathcal{M}|})$. Therefore, we get an integral allocation, $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|})$ such that $v_i(A_i) \geq V_i(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|})) - v_i(\ell_i)$ where $\ell_i \coloneqq \operatorname{argmax}_j v_i(\{j\})$. Since by assumption $v_i(\ell_i) \leq \epsilon\mathsf{MMS}_i$ by assumption, and from Claims 24 and 25, $V_i(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|})) \geq \left(1 - \frac{1}{e}\right) \mathsf{MMS}_i$, the theorem follows. ◀

Algorithm 4 gives a ≈ 0.316-approximation for MMS with submodular valuations. To analyze this, we first understand the algorithm. Step (6) is performing our NonUniformPipageRounding. The while loop in lines (2) to (4) performs an operation that is called as "single good reduction". The following claim regarding this is well-known and used in almost all works on MMS.

▷ Claim 27. Given a fair division instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$, the MMS value of any agent is retained if we remove any single agent and any single good. That is,

$$\mathsf{MMS}_i^{|\mathcal{N}|}(\mathcal{M}) \leq \mathsf{MMS}_i^{|\mathcal{N}|-1}(\mathcal{M} \setminus \{g\}) \qquad \text{for all } g \in \mathcal{M}$$

We can now prove the following result for Algorithm 4.

Algorithm 4 $\frac{1}{2}(1-\frac{1}{e}-o(1))$-approximation for MMS with submodular valuations.

Input : Instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$
Output : An allocation where each agent receives at least $\frac{1}{2}(1-\frac{1}{e}-o(1))\mathsf{MMS}_i$.

1 Initialize $\mathcal{N}' \leftarrow \mathcal{N}$ and $\mathcal{M}' \leftarrow \mathcal{M}$
2 **while** *there exists* $i \in \mathcal{N}'$ *and* $g \in \mathcal{M}'$ *with* $v_i(g) \geq \frac{1}{2}V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|})$ **do**
3 Allocate g to i
4 Update $\mathcal{N}' \leftarrow \mathcal{N}' \setminus i$ and $\mathcal{M}' \leftarrow \mathcal{M}' \setminus g$
5 Initialize $\mathbf{x} = \left(\frac{1}{|\mathcal{N}'|}(\mathbf{1}^{|\mathcal{N}'|\times|\mathcal{M}'|})\right)$
6 $(A_1, \ldots, A_{|\mathcal{N}'|}) \leftarrow$ NonUniformPipageRound($\mathbf{x}$)
7 **return** $(A_1, \ldots, A_{|\mathcal{N}|})$

▶ **Theorem 28.** *Given a fair division instance,* $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ *with submodular valuations, Algorithm 4 outputs an allocation where each agent receives a value of* $\frac{1}{2}\left(1-\frac{1}{e}-o(1)\right)\mathsf{MMS}_i$.

Proof. First, let us assume that we can compute $V_i(\mathbf{y})$ exactly for any $\mathbf{y} \in [0,1]^{|\mathcal{N}|\times|\mathcal{M}|}$. Note that as per Algorithm 4, an agent can receive an allocation either in the while loop of Steps 2-4 or in Step 6 via NonUniformPipageRounding. In every iteration of the while loop, we remove one agent and one good. Therefore, by Claim 27, the MMS value of any agent in the residual instance does not decrease. As a result, until the algorithm reaches Step 6, we always have $\mathsf{MMS}_i \leq (\frac{e}{e-1})V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|})$. Therefore, any agent who gets a good in the while loop receives a good of value at least $\frac{1}{2}(1-\frac{1}{e})\mathsf{MMS}_i$. Now, in Step 6, by Lemma 8, we get an allocation of value at least $V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|}) - v_i(\ell_i) \geq \frac{1}{2}V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|}) \geq \frac{1}{2}(1-\frac{1}{e})\mathsf{MMS}_i$.

Now, we note that the values of $V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|})$ are computed by sampling and in polynomial time, we can only compute them to a factor of $(1-\epsilon)$. Accounting for this, we get that in randomized polynomial time we can guarantee each agent a value of $(1-\frac{1}{e}-o(1))\mathsf{MMS}_i$. This proves the theorem. ◀

References

1 Hannaneh Akrami and Jugal Garg. Breaking the 3/4 barrier for approximate maximin share. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 74–91. SIAM, 2024. doi:10.1137/1.9781611977912.4.

2 Hannaneh Akrami, Jugal Garg, Eklavya Sharma, and Setareh Taki. Improving approximation guarantees for maximin share. In *25th ACM Conference on Economics and Computation*, pages 198–198. ACM, 2024.

3 Georgios Amanatidis, Evangelos Markakis, Afshin Nikzad, and Amin Saberi. Approximation algorithms for computing maximin share allocations. *ACM Transactions on Algorithms (TALG)*, 13(4):1–28, 2017. doi:10.1145/3147173.

4 Nima Anari, Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. Nash social welfare, matrix permanent, and stable polynomials. *arXiv preprint arXiv:1609.07056*, 2016. arXiv:1609.07056.

5 Nima Anari, Tung Mai, Shayan Oveis Gharan, and Vijay V. Vazirani. Nash social welfare for indivisible items under separable, piecewise-linear concave utilities. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2274–2290. SIAM, 2018. doi:10.1137/1.9781611975031.147.

6 Etienne Bamas, Paritosh Garg, and Lars Rohwedder. The Submodular Santa Claus Problem in the Restricted Assignment Case. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:18, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2021.22.

7 Étienne Bamas, Sarah Morell, and Lars Rohwedder. The submodular Santa Claus problem. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 616–640. SIAM, 2025. doi:10.1137/1.9781611978322.19.

8 Nikhil Bansal and Maxim Sviridenko. The santa claus problem. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 31–40, 2006. doi:10.1145/1132516.1132522.

9 Siddharth Barman and Sanath Kumar Krishnamurthy. Approximation algorithms for maximin fair division. *ACM Transactions on Economics and Computation (TEAC)*, 8(1):1–28, 2020. doi:10.1145/3381525.

10 Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Finding fair and efficient allocations. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 557–574, 2018. doi:10.1145/3219166.3219176.

11 Xiaohui Bei, Yuda Feng, Yang Hu, Shi Li, and Ruilong Zhang. Nash social welfare with submodular valuations: Approximation algorithms and integrality gaps. *arXiv preprint arXiv:2504.09669*, 2025. To appear in Proc. of ACM STOC 2026. doi:10.48550/arXiv.2504.09669.

12 Ivona Bezáková and Varsha Dani. Allocating indivisible goods. *ACM SIGecom Exchanges*, 5(3):11–18, 2005. doi:10.1145/1120680.1120683.

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

14 Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 182–196. Springer, 2007.

15 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. doi:10.1137/080733991.

16 Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, 7(3):1–32, 2019. doi:10.1145/3355902.

17 Deeparnab Chakrabarty, Julia Chuzhoy, and Sanjeev Khanna. On allocating goods to maximize fairness. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 107–116. IEEE, 2009. doi:10.1109/FOCS.2009.51.

18 Chandra Chekuri, T.S. Jayram, and Jan Vondrak. On multiplicative weight updates for concave and submodular function maximization. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS '15, pages 201–210, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2688073.2688086.

19 Chandra Chekuri, Pooja Kulkarni, Rucha Kulkarni, and Ruta Mehta. 1/2-approximate mms allocation for separable piecewise linear concave valuations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38(9), pages 9590–9597, 2024. doi:10.1609/AAAI.V38I9.28815.

20 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. doi:10.1109/FOCS.2010.60.

21 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 783–792, 2011. doi:10.1145/1993636.1993740.

22 Siu-Wing Cheng and Yuchen Mao. Restricted Max-Min Allocation: Approximation and Integrality Gap. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.38.

23 Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay V Vazirani, and Sadra Yazdanbod. Convex program duality, fisher markets, and nash social welfare. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 459–460, 2017. doi:10.1145/3033274.3085109.

24 Richard Cole and Vasilis Gkatzelis. Approximating the nash social welfare with indivisible items. *SIAM Journal on Computing*, 47(3):1211–1236, 2018. doi:10.1137/15M1053682.

25 Shahar Dobzinski and Michael Schapira. An improved approximation algorithm for combinatorial auctions with submodular bidders. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1064–1073, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109675.

26 Yuda Feng, Yang Hu, Shi Li, and Ruilong Zhang. Constant approximation for weighted nash social welfare with submodular valuations. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1395–1405, 2025. doi:10.1145/3717823.3718203.

27 Yuda Feng and Shi Li. A note on approximating weighted nash social welfare with additive valuations. *TheoretiCS*, 4, 2025. doi:10.46298/THEORETICS.25.17.

28 Jugal Garg, Martin Hoefer, and Kurt Mehlhorn. Approximating the nash social welfare with budget-additive valuations. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2326–2340. SIAM, 2018. doi:10.1137/1.9781611975031.150.

29 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*, pages 1298–1310, 2023.

30 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*, pages 1412–1425, 2021.

31 Jugal Garg, Edin Husić, Wenzheng Li, László A. Végh, and Jan Vondrák. Approximating nash social welfare by matching and local search, 2026. arXiv:2211.03883.

32 Jugal Garg, Pooja Kulkarni, and Rucha Kulkarni. Approximating nash social welfare under submodular valuations through (un) matchings. *ACM Transactions on Algorithms*, 19(4):1–25, 2023. doi:10.1145/3613452.

33 Jugal Garg, Peter McGlaughlin, and Setareh Taki. Approximating Maximin Share Allocations. In *2nd Symposium on Simplicity in Algorithms (SOSA 2019)*, volume 69 of *Open Access Series in Informatics (OASIcs)*, pages 20:1–20:11. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/OASIcs.SOSA.2019.20.

34 Jugal Garg and Setareh Taki. An improved approximation algorithm for maximin shares. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 379–380, 2020. doi:10.1145/3391403.3399526.

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

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

37 Euiwoong Lee. Apx-hardness of maximizing nash social welfare with indivisible items. *Inf. Process. Lett.*, 122:17–20, 2017. doi:10.1016/J.IPL.2017.01.012.

38 Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 18–28, 2001. doi:10.1145/501158.501161.

39 Jan Karel Lenstra, David B Shmoys, and Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical programming*, 46:259–271, 1990. doi:10.1007/BF01585745.

40 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. IEEE, 2022.

41 Li Lijun, Xu Chenyang, Yang Liuyi, and Ruilong Zhang. Fair submodular maximization over a knapsack constraint. In *34th International Joint Conference on Artificial Intelligence*, 2025.

42 Richard J Lipton, Evangelos Markakis, Elchanan Mossel, and Amin Saberi. On approximately fair allocations of indivisible goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, pages 125–131, 2004. doi:10.1145/988772.988792.

43 Vahab Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proceedings of the 9th ACM conference on Electronic commerce*, pages 70–77, 2008. doi:10.1145/1386790.1386805.

44 George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978. doi:10.1007/BF01588971.

45 Serge A Plotkin, David B Shmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20(2):257–301, 1995. doi:10.1287/MOOR.20.2.257.

46 Ariel D Procaccia and Junxing Wang. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 675–692, 2014. doi:10.1145/2600057.2602835.

47 David B Shmoys and Éva Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical programming*, 62(1):461–474, 1993. doi:10.1007/BF01585178.

48 Gilad Ben Uziahu and Uriel Feige. On fair allocation of indivisible goods to submodular agents. *arXiv preprint arXiv:2303.12444*, 2023. doi:10.48550/arXiv.2303.12444.

49 Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 67–74, 2008. doi:10.1145/1374376.1374389.