Algorithms and Complexity Group
Institute for Computer Science
Goethe University Frankfurt

Summer Semester 2023

Seminar Approximation Algorithms

Approximating Nash Social Welfare under Submodular Valuations through (Un)Matchings

Based on the paper of the same name by Garg, Kulkarni and Kulkarni [15].

Zeno Adrian Weil

16th June 2023

Examiner: Prof Dr Martin Hoefer
Supervisor: Dr Giovanna Varricchio

Abstract

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

Todo list

1 Introduction

The study of distributing resources amongst one or more receivers is an interdisciplinary field and is interesting from both a computational (how to find an allocation) and a qualitative (what a good allocation makes) standpoint [8]. Its areas of application are manifold: industrial procurement, where the preferences of buyers and sellers need be appropriately captured and real-world constraints on goods and services be taken into account [8]; mobile edge computing, where computation and storage are taken on by physically close cloud systems but participation has to be incentivised [2, 11]; manufacturing processes, where tasks should be scheduled efficiently within and between many production sites and disturbances be quickly paid heed to [8]; water management, where hostile countries must come to mutual agreements on the withdrawal from contested rivers [10].

In this seminar paper, we focus on unsharable and indivisible resources, which we term *items*. The receivers of those items are called *agents*. The distributions of items amongst agents are modelled through allocations.

Definition 1. Let $\mathcal{G}$ be a set of m items and $\mathcal{A}$ be a set of n agents. An *allocation* is a tuple $\boldsymbol{x} = (\boldsymbol{x}_i)_{i \in \mathcal{A}}$ of *bundles* $\boldsymbol{x}_i \subset \mathcal{G}$ such that each item is element of exactly one bundle, that is, $\bigcup_{i \in \mathcal{A}} \boldsymbol{x}_i = \mathcal{G}$ and $\boldsymbol{x}_i \cap \boldsymbol{x}_{i'} = \emptyset$ for all $i \neq i'$. An item $j \in \mathcal{G}$ is *assigned* to agent $i \in \mathcal{A}$ if $j \in \boldsymbol{x}_i$ holds. or rather 'alloc-ated'?

The satisfaction of an agent i with her bundle $\boldsymbol{x}_i$ is measured by her *valuation function* v_i, which assigns each set of items a real value. We always assume that valuation functions are non-negative, i. e., $v_i(\mathcal{S}) \geq 0 \ \forall \mathcal{S} \subset \mathcal{G}$, monotonically non-decreasing, i. e., $v_i(\mathcal{S}_1) \leq v_i(\mathcal{S}_2)$ $\forall \mathcal{S}_1 \subset \mathcal{S}_2 \subset \mathcal{G}$, and normalised, i. e., $v_i(\emptyset) = 0$. Besides fulfilling these properties, the valuation functions can come from a plethora of function families. We will discuss additive and submodular functions in greater detail.

Additive The valuation $v_i(\mathcal{S})$ of an agent i for a set $\mathcal{S} \subset \mathcal{G}$ of items j is the sum of individual valuations $v_i(j)$, that is, $v_i(\mathcal{S}) = \sum_{j \in \mathcal{S}} v_i(j)$.

These are fairly simple but useful functions, and many expansions exist. [13, 15, p. 3]

Submodular Let $v_i(\mathcal{S}_1 \mid \mathcal{S}_2) := v_i(\mathcal{S}_1 \cup \mathcal{S}_2) - v_i(\mathcal{S}_2)$ denote the marginal valuation of agent i or rather utility? for a set $\mathcal{S}_1 \subset \mathcal{G}$ of items over a disjoint set $\mathcal{S}_2 \subset \mathcal{G}$. This valuation function satisfies the submodularity constraint $v_i(j \mid \mathcal{S}_1 \cup \mathcal{S}_2) \leq v_i(j \mid \mathcal{S}_1)$ for all agents $i \in \mathcal{A}$, items $j \in \mathcal{G}$ and sets $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{G}$ of items.

Submodular valuation functions (which encompass additive ones) have the property that the gain from assigning new items is smaller, the bigger the bundles are. Diminishing returns are a common phenomenon in economics, making submodular functions worthwhile to study. [20]

In a slight abuse of notation, we sometimes omit curly braces delimiting a set, so we write $v_i(j_1, j_2, \dots)$ but mean $v_i(\{j_1, j_2, \dots\})$ for example.

In order to measure and maximise the overall satisfaction of all agents, one needs to combine their valuations. Several options arise here; common choices are the utilitarian social welfare (USW), that is the sum of all valuations [2, 8, 10, 15, 20], and the egalitarian social welfare (ESW), that is the minimum of all valuations [8, 15]. We consider a third one, the Nash social welfare (NSW).

Problem 1. Given a set $\mathcal{G}$ of items and a set $\mathcal{A}$ of agents with monotonically non-decreasing valuation functions $v_i \colon \mathcal{P}(\mathcal{G}) \to \mathbb{R}_0^+$ and agent weights $\eta_i \in \mathbb{R}^+$ for all agents $i \in \mathcal{A}$, the

Nash social welfare problem is to find an allocation $\boldsymbol{x}^*$ maximising the weighted geometric mean of valuations, that is,

$$\boldsymbol{x}^* \stackrel{!}{=} \arg\max_{\boldsymbol{x} \in \boldsymbol{X}_{\mathcal{A}}(\mathcal{G})} \{\mathrm{NSW}(\boldsymbol{x})\} \quad \text{with } \mathrm{NSW}(\boldsymbol{x}) := \Big(\prod_{i \in \mathcal{A}} v_i(\boldsymbol{x}_i)^{\eta_i}\Big)^{1/\sum_{i\in\mathcal{A}} \eta_i}$$

where $\boldsymbol{X}_{\mathcal{A}}(\mathcal{G})$ is the set of all possible allocations. The problem is called *symmetric* if all agent weights η_i are equal, and *asymmetric* otherwise.

For the techniques employed in later sections, it is beneficial to consider the logarithmic NSW, that is,

$$\log \mathrm{NSW}(\boldsymbol{x}) = \frac{1}{\sum_{i\in\mathcal{A}} \eta_i} \cdot \sum_{i\in\mathcal{A}} \eta_i \log v_i(\boldsymbol{x}_i), \tag{1}$$

which is a sum instead of a product. The NSW strikes as middle ground between the USW and the ESW, which focus on efficiency (height of overalls satisfaction) and fairness (how agents value other agents' bundles), respectively. In addition, it exhibits scale-freeness, that is invariance to the scales in which each valuation is expressed, rendering e. g. relative instead of absolute valuations unproblematic. Even though the NSW problem is $\mathcal{NP}$-hard and $\mathcal{APX}$-hard, approximate solutions largely keep the properties of optimal allocations. [3, 6, 9, 22, cf. remark 2] We use the following definition for the approximation factor.

Definition 2. An algorithm for a maximisation problem is α-approximative if, for every problem instance I and output $\mathrm{ALG}(I)$, it holds $\mathrm{ALG}(I) \geq \mathrm{OPT}(I)/\alpha$, where $\mathrm{OPT}(I)$ is the optimal value.

rephrase

check if asymmetric

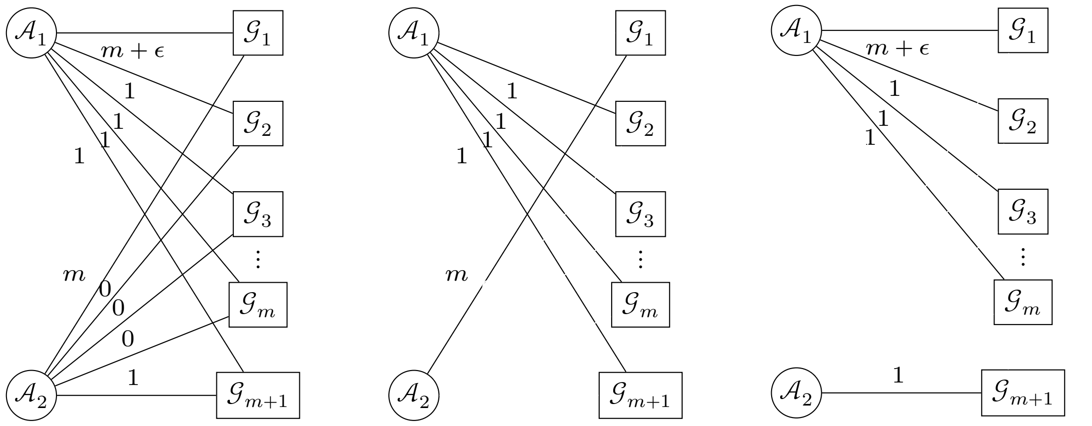
As reference, a quick overview of the research situation[1] of the USW and ESW: For the submodular USW, a lower bound of $\frac{e}{e-1}$ on the approximation factor was proven [20] and an approximation algorithm achieving said factor shown [24] — the additive USW is trivially solvable through repeated maximum matchings. For the additive ESW, a randomised $(320\sqrt{n}\log^3 n)$-approximative algorithm employing linear programming [1] and a hardness of 2 [5] are know. An $(2n-1)$-approximative algorithm exists for the submodular ESW [19].

In contrast, the NSW is less well understood[1]. A 1.45-approximative algorithm is known for the symmetric additive NSW [3]. For the symmetric submodular NSW, a $(m-n+1)$-approximative algorithm has been devised [22]. Both approaches exploit the symmetry of the studied problem and fail to be extended to the asymmetric case. Moreover, an approximation factor dependant on the number of items is not desirable as the number of items vastly exceeds the number of agents in many applications.

Garg, Kulkarni and Kulkarni [15] fill this knowledge gap by providing two algorithms with polynomial runtime. The first one, *SMatch*, computes an allocation for the asymmetric additive NSW and is $2n$-approximative. It does so by *s*martly *match*ing agents and items in a bipartite graph. The second one, *RepReMatch*, computes an allocation for the asymmetric submodular NSW and is $(2n(\log_2 n + 3))$-approximative. It does so by *rep*eatedly computing *match*ings, which then get partly annulled, so that items can be *re*matched. We present and analyse both algorithms in section 2 and section 3 of our seminar report, respectively. In section 4, we analyse the hardness of the submodular NSW. Section 5 comprises the conclusion, a summary of newly published work since 2020, and an outlook on open questions.

[1]The overview is given as it was roughly at the end of the year 2019, when Garg, Kulkarni and Kulkarni wrote their paper [15] on which this seminar report is based.

- matroids? preliminaries [14]
- exponentially many subsets; additive vs. submodular?
- hardness; constant additive in 2015; $\mathcal{O}(n)$ for sym. subadditive but hard to improve for special cases; unbounded integrality grap [14]



(a) problem instance **(b)** OPT $= \sqrt{m \cdot m}$ **(c)** ALG $\leq \sqrt{(2m + \epsilon - 1) \cdot 1}$

Figure 1: An example with two symmetric agents showing that simple, repeated matching without consideration of the future leads to an approximation factor dependent on the number of items. Agent $\mathcal{A}_1$ values item $\mathcal{G}_1$ at $m + \epsilon$ and all other items at 1. Agent $\mathcal{A}_2$ values item $\mathcal{G}_1$ at m, item $\mathcal{G}_{m+1}$ at 1 and all other items at 0. In an optimal allocation, item $\mathcal{G}_1$ would be assigned to agent $\mathcal{A}_2$ and all other items to agent $\mathcal{A}_1$, resulting in a NSW of $\sqrt{m \cdot m} = m$. A repeated maximum matching algorithm would greedily assign item $\mathcal{G}_1$ to agent $\mathcal{A}_1$ and item $\mathcal{G}_{m+1}$ to agent $\mathcal{A}_2$ in the first round. Even if all remaining items were going to be assigned to agent $\mathcal{A}_1$, the NSW will never surpass $\sqrt{(2m + \epsilon - 1) \cdot 1} < \sqrt{2m}$. The approximation factor $\alpha \approx \sqrt{m/2}$ therefore depends on the number of items.

2 SMatch

In the case of an equal number of agents and items, i. e., $n = m$, the *additive* NSW can be solved exactly by finding a maximum matching on a bipartite graph with the sets of agents and of items as its parts; as weight of the edge between agent i and item j, use $\eta_i \log v_i(j)$, that is the weighted valuation of item j by agent i in the logarithmic Nash social welfare. Should there be more items than agents, then it would be obvious at first to just repeatedly find a maximum matching and assign the items accordingly until all items are assigned. The flaw of this idea is that such a greedy algorithm only considers the valuations of items in the current matching and perhaps the valuations of items already assigned. As the example in fig. 1 demonstrates, this leads to an algorithm with an approximation factor dependent on the number m of items. The geometric mean of the NSW favours allocations with similarly valued bundles, wherefore it may be beneficial to give items to agents who cannot expect many more valuable items in the future instead of to agents who value the item a bit more but do so for other items as well.

The algorithm SMatch, described in algorithm 1, eliminates the flaw by first gaining foresight of the valuations of items assigned after the first matching, achieving an approximation factor of $2n$ (cf. theorem 1 later on). For a fixed agent i, order the items in descending order of valuations and denote the j-th most liked item by $\mathcal{G}_i^j$. To obtain a well-defined order, items of equal rank are further ordered numerically. SMatch does in fact repeatedly match items. During the first matching, however, the edge weights are defined as $\eta_i \log(v_i(j) + u_i/n)$ for an edge between agent i and item j. The addend u_i serves as estimation of the valuation of items assigned after the first matching (cf. lemma 2) and is defined as

$$u_i := \min_{\substack{\mathcal{S} \subset \mathcal{G} \\ |\mathcal{S}| \leq 2n}} \{v_i(\mathcal{G} \setminus \mathcal{S})\} = v_i(\mathcal{G}_i^{2n+1}, \ldots, \mathcal{G}_i^m). \tag{2}$$

Algorithm 1: SMatch for the asymmetric additive NSW

Input : set $\mathcal{G}$ of m items, set $\mathcal{A}$ of n agents, additive valuation functions $v_i\colon \mathcal{P}(\mathcal{G}) \to \mathbb{R}_0^+$ and weights $\eta_i \in \mathbb{R}^+$ for all agents $i \in \mathcal{A}$

Output: $2n$-approximation $\boldsymbol{x} = (\boldsymbol{x}_i)_{i\in\mathcal{A}}$ of an optimal allocation

$\boldsymbol{x}_i \leftarrow \emptyset \quad \forall i \in \mathcal{A}$

$u_i \leftarrow v_i(\mathcal{G}_i^{2n+1}, \dots, \mathcal{G}_i^{m}) \quad \forall i \in \mathcal{A}$ ▷ *estimation of future valuations*

$\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(j) + u_i/n) \mid i \in \mathcal{A}, j \in \mathcal{G} \,\}$ ▷ *edge weights*

$G \leftarrow (\mathcal{A}, \mathcal{G}, \mathcal{W})$ ▷ *bipartite graph*

$\mathcal{M} \leftarrow \mathrm{max\_weight\_matching}(G)$

$\boldsymbol{x}_i \leftarrow \{\, j \mid (i,j) \in \mathcal{M} \,\} \quad \forall i \in \mathcal{A}$ ▷ *assign according to matching*

$\mathcal{G}^{\mathrm{rem}} \leftarrow \mathcal{G} \setminus \{\, j \mid (i,j) \in \mathcal{M} \,\}$ ▷ *remove assigned items*

while $\mathcal{G}^{\mathrm{rem}} \neq \emptyset$ **do**

  $\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(j) + v_i(\boldsymbol{x}_i)) \mid i \in \mathcal{A}, j \in \mathcal{G}^{\mathrm{rem}} \,\}$

  $G \leftarrow (\mathcal{A}, \mathcal{G}^{\mathrm{rem}}, \mathcal{W})$

  $\mathcal{M} \leftarrow \mathrm{max\_weight\_matching}(G)$

  $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i \cup \{\, j \mid (i,j) \in \mathcal{M} \,\} \quad \forall i \in \mathcal{A}$

  $\mathcal{G}^{\mathrm{rem}} \leftarrow \mathcal{G}^{\mathrm{rem}} \setminus \{\, j \mid (i,j) \in \mathcal{M} \,\}$

end while

return $\boldsymbol{x}$

The set $\mathcal{S}$ has less than $2n$ elements only if there are less than $2n$ items in total. From the second matching onwards, the edge weights are defined as $\eta_i \log(v_i(j) + v_i(\boldsymbol{x}_i))$, where $\boldsymbol{x}_i$ is the continuously updated bundle of agent i. The addend $v_i(\boldsymbol{x}_i)$ could lead to better allocations in applications, but does not improve the approximation factor asymptotically.

To calculate the approximation factor of SMatch, we first need to establish a lower bound on the valuation of single items. For convenience, we order the items in the final bundle $\boldsymbol{x}_i = \{h_i^1, \dots, h_i^{\tau_i}\}$ of agent i by the order in which they were assigned, so that item h_i^t is assigned according to the t-th matching. Note that it holds $v_i(h_i^t) \geq v_i(h_i^{t'})$ for all $t' \geq t$.

Lemma 1. *For each agent $i \in \mathcal{A}$, her final bundle $\boldsymbol{x}_i = \{h_i^1, \dots, h_i^{\tau_i}\}$, and her tn-th most highly valued item $\mathcal{G}_i^{tn}$, it holds $v_i(h_i^t) \geq v_i(\mathcal{G}_i^{tn})$ for all $t = 1, \dots, \tau_i$.*

Proof. Before the t-th matching, no more than $(t-1)n$ items out of the tn most highly valued items $\mathcal{G}_i^1, \dots, \mathcal{G}_i^{tn}$ have been assigned in previous matchings since at most n many out of those items are assigned each time. Because of the t-th matching, at most $n-1$ more could be assigned to all other agents $i' \neq i$, leaving at least one item of $\mathcal{G}_i^1, \dots, \mathcal{G}_i^{tn}$ unassigned. Since $v_i(\mathcal{G}_i^k) \geq v_i(\mathcal{G}_i^{tn})$ for all $k \leq tn$ by definition of $\mathcal{G}_i^{\cdot}$, the lemma follows. ◻

We can now establish $u_i/n = v_i(\mathcal{G}_i^{2n+1}, \dots, \mathcal{G}_i^m)/n$ as lower bound on the valuations of items assigned after the first matching.

Lemma 2. *For each agent $i \in \mathcal{A}$ and her final bundle $\boldsymbol{x}_i = \{h_i^1, \dots, h_i^{\tau_i}\}$, it holds $v_i(h_i^2, \dots, h_i^{\tau_i}) \geq u_i/n$.*

Proof. By lemma 1 and definition of $\mathcal{G}_i^{\cdot}$, every item h_i^t is worth at least as much as each item $\mathcal{G}_i^{tn+k}$ with $k \in \{1, \dots, n\}$ and, consequently, its valuation $v_i(h_i^t)$ is at least as high as the mean valuation $\frac{1}{n} v_i(\mathcal{G}_i^{tn+1}, \dots, \mathcal{G}_i^{tn+n})$. Further, it holds $\tau_i n + n \geq m$ since agent i gets assigned $\tau_i \geq \lfloor \frac{m}{n} \rfloor \geq \frac{m}{n} - 1$ many items. Together, this yields

i: I do not get the reason for the extra step in the original paper.

$$v_i(h_i^2, \dots, h_i^{\tau_i}) = \sum_{t=2}^{\tau_i} v_i(h_i^t) \geq \sum_{t=2}^{\tau_i} \frac{1}{n} v_i(\mathcal{G}_i^{tn+1}, \dots, \mathcal{G}_i^{tn+n}) \geq \frac{1}{n} v_i(\mathcal{G}_i^{2n+1}, \dots, \mathcal{G}_i^{m}) = \frac{u_i}{n}. \quad (3)$$

$\square$

Remark 1. In lemma 2, we assumed non-zero valuations for all items, hence the bundle lengths of $\tau_i \geq \lfloor \frac{m}{n} \rfloor$. Of course in an actual program, one would not assign items to agents who value them at zero. Nevertheless, lemma 2 still holds inasmuch as additional zero valuations in eq. (3) do not change the sum.

This allows us to calculate an approximation factor for SMatch by comparing its output with an optimal allocation $\boldsymbol{x}^*$.

Theorem 1. *SMatch is $2n$-approximative.*

Proof. Lemma 2 can be plugged into the logarithmic NSW:

$$\log \mathrm{NSW}(\boldsymbol{x}) = \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log v_i(h_i^1, \dots, h_i^{\tau_i}) \tag{4}$$

$$= \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log(v_i(h_i^1) + v_i(h_i^2, \dots, h_i^{\tau_i})) \tag{5}$$

$$\geq \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log(v_i(h_i^1) + u_i/n) \tag{6}$$

Notice that the first matching of SMatch maximises the sum in eq. (6). Thus, assigning all agents i their respective most highly valued item g_i^1 in an optimal bundle $\boldsymbol{x}_i^* = \{g_i^1, \dots, g_i^{\tau_i^*}\}$ yields the even lower bound

$$\log \mathrm{NSW}(\boldsymbol{x}) \geq \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log(v_i(g_i^1) + u_i/n). \tag{7}$$

Recall the definition of u_i from eq. (2). Consider a slightly modified variant:

$$u_i = v_i(\mathcal{G} \setminus \mathcal{S}_i) \text{ with } \mathcal{S}_i := \underset{\substack{\mathcal{S} \subset \mathcal{G} \\ |\mathcal{S}| \leq 2n}}{\arg\min} \{v_i(\mathcal{G} \setminus \mathcal{S})\} \tag{8}$$

Moreover, consider the set $\mathcal{S}_i^*$ of the (at most) $2n$ most highly valued items in the optimal bundle $\boldsymbol{x}_i^*$, i. e.

$$\mathcal{S}_i^* := \underset{\substack{\mathcal{S} \subset \mathcal{G} \\ |\mathcal{S}| \leq 2n}}{\arg\min} \{v_i(\boldsymbol{x}_i^* \setminus \mathcal{S})\}. \tag{9}$$

It holds $v_i(g_i^1) \geq \frac{1}{2n} v_i(\mathcal{S}_i^*)$ because of $v_i(g_i^1) \geq v_i(j)$ for all $j \in \mathcal{S}_i^*$. Furthermore, it holds $u_i = v_i(\mathcal{G} \setminus \mathcal{S}_i) \geq v_i(\boldsymbol{x}_i^* \setminus \mathcal{S}_i) \geq v_i(\boldsymbol{x}_i^* \setminus \mathcal{S}_i^*)$. We can insert these two inequalities into eq. (7) and prove the theorem thereby:

$$\log \mathrm{NSW}(\boldsymbol{x}) \geq \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log\left(\frac{v_i(\mathcal{S}_i^*)}{2n} + \frac{v_i(\boldsymbol{x}_i^* \setminus \mathcal{S}_i^*)}{n}\right) \tag{10}$$

$$\geq \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log\left(\frac{v_i(\boldsymbol{x}_i^*)}{2n}\right) \tag{11}$$

$$= \log\left(\frac{\mathrm{NSW}(\boldsymbol{x}^*)}{2n}\right) \tag{12}$$

$\square$

The analysis is asymptotically tight. It is possible to design an instance for the asymmetric NSW such that SMatch achieves an approximation ratio approaching $2/n$. It remains to be shown whether the symmetric NSW is equally hard. [15, Section 6.3]

Remark 2. SMatch produces *fair* allocations which are *envy-free up to one item* (EF1). An allocation $\boldsymbol{x}$ is EF1 if, for every pair $(i_1, i_2) \in \mathcal{A}^2$ of agents, one needs to remove at most one item from the bundle $\boldsymbol{x}_{i_2}$ of agent i_2 so that agent i_1 does not want to swap bundles. In other words, either it holds $v_{i_1}(\boldsymbol{x}_{i_1}) \geq v_{i_2}(\boldsymbol{x}_{i_2})$ or there is an item $j \in \boldsymbol{x}_{i_2}$ such that $v_{i_1}(\boldsymbol{x}_{i_1}) \geq v_{i_2}(\boldsymbol{x}_{i_2} \setminus \{j\})$. [15, Section 5.2]

Pigou-Dalton-Prinzip?

PO?

3 RepReMatch

The algorithm SMatch estimates the valuation of the lowest-value items by determining the set of highest-value items and then valuing the remaining items. Unfortunately, this approach does not work for general submodular valuation functions because taking the set of highest-value items away does not necessarily leave a set of lowest-value items. In fact, it can be shown [23] that determining the set of lowest-value items is approximable only within a factor of $\Omega(\sqrt{m/\ln m})$.

For this reason, the algorithm RepReMatch, described in algorithm 2, relies on an approach with three phases, achieving an approximation factor of $2n(\log_2 n + 3)$ (cf. theorem 2). In phase I, a sufficiently big set of high-value items is determined through repeated matchings. This phase serves merley to determine this set, so items are assigned temporarily only. The edge weights reflect this by taking the valuations of just single items

Algorithm 2: RepReMatch for the asymmetric submodular NSW

Input : set $\mathcal{G}$ of m items, set $\mathcal{A}$ of n agents, additive valuation functions $v_i \colon \mathcal{P}(\mathcal{G}) \to \mathbb{R}_0^+$ and weights $\eta_i \in \mathbb{R}^+$ for all agents $i \in \mathcal{A}$

Output: $2n(\log_2 n + 3)$-approximation $\boldsymbol{x}^{\text{III}} = (\boldsymbol{x}_i^{\text{III}})_{i \in \mathcal{A}}$ of an optimal allocation

Phase I:

$\boldsymbol{x}_i^{\text{I}} \leftarrow \emptyset \quad \forall i \in \mathcal{A}$

$\mathcal{G}^{\text{rem}} \leftarrow \mathcal{G}$

for $t \leftarrow 1, \dots, \lceil \log_2 n \rceil + 1$ **do**

    if $\mathcal{G}^{\text{rem}} \neq \emptyset$ **then**

        $\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(j)) \mid i \in \mathcal{A}, j \in \mathcal{G}^{\text{rem}} \,\}$ ▷ *valuation of single item*

        $G \leftarrow (\mathcal{A}, \mathcal{G}^{\text{rem}}, \mathcal{W})$

        $\mathcal{M} \leftarrow \text{max\_weight\_matching}(G)$

        $\boldsymbol{x}_i^{\text{I}} \leftarrow \boldsymbol{x}_i^{\text{I}} \cup \{j\} \quad \forall (i,j) \in \mathcal{M}$

        $\mathcal{G}^{\text{rem}} \leftarrow \mathcal{G}^{\text{rem}} \setminus \{\, j \mid (i,j) \in \mathcal{M} \,\}$

    end if

end for

Phase II:

$\boldsymbol{x}_i^{\text{II}} \leftarrow \emptyset \quad \forall i \in \mathcal{A}$ ▷ *put allocation $\boldsymbol{x}^{\text{I}}$ away and start a new one*

while $\mathcal{G}^{\text{rem}} \neq \emptyset$ **do**

    $\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(\boldsymbol{x}_i^{\text{II}} \cup \{j\})) \mid i \in \mathcal{A}, j \in \mathcal{G}^{\text{rem}} \,\}$ ▷ *val. of item & cur. bundle*

    $G \leftarrow (\mathcal{A}, \mathcal{G}^{\text{rem}}, \mathcal{W})$

    $\mathcal{M} \leftarrow \text{max\_weight\_matching}(G)$

    $\boldsymbol{x}_i^{\text{II}} \leftarrow \boldsymbol{x}_i^{\text{II}} \cup \{j\} \quad \forall (i,j) \in \mathcal{M}$

    $\mathcal{G}^{\text{rem}} \leftarrow \mathcal{G}^{\text{rem}} \setminus \{\, j \mid (i,j) \in \mathcal{M} \,\}$

end while

Phase III:

$\mathcal{G}^{\text{rem}} \leftarrow \bigcup_{i \in \mathcal{A}} \boldsymbol{x}_i^{\text{I}}$ ▷ *release items assigned in phase I*

$\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(\boldsymbol{x}_i^{\text{II}} \cup \{j\})) \mid i \in \mathcal{A}, j \in \mathcal{G}^{\text{rem}} \,\}$ ▷ *val. of item & cur. bundle*

$G \leftarrow (\mathcal{A}, \mathcal{G}^{\text{rem}}, \mathcal{W})$

$\mathcal{M} \leftarrow \text{max\_weight\_matching}(G)$

$\boldsymbol{x}_i^{\text{III}} \leftarrow \boldsymbol{x}_i^{\text{II}} \cup \{j\} \quad \forall (i,j) \in \mathcal{M}$

$\mathcal{G}^{\text{rem}} \leftarrow \mathcal{G}^{\text{rem}} \setminus \{\, j \mid (i,j) \in \mathcal{M} \,\}$

$\boldsymbol{x}^{\text{III}} \leftarrow \text{arbitrary\_allocation}(\mathcal{A}, \mathcal{G}^{\text{rem}}, \boldsymbol{x}^{\text{III}}, (v_i)_{i \in \mathcal{A}})$

return $\boldsymbol{x}^{\text{III}}$

---

into account.

In phase II, the remaining items are assigned normally through repeated matchings. Consequently, each edge weight is updated in each round to be the weighted logarithm of the valuation of both the respective item and the items assigned so far.

In phase III, the high-value items assigned in phase I are released. With the knowledge of items assigned in phase II, one maximum weight matching is calculated, and the matched items are assigned accordingly. Again each edge weight is the weighted logarithm of the valuation of both the respective item and the respective agent's bundle from phase II. The remaining released items are assigned arbitrarily.

We start by analysing phase II as it is the first phase with definitive assignments. To this end, we introduce two types of item sets. Note that we use the term *round* to refer to the iterations of the loops in the phases I and II. For ease of notation, we refer to the moment right before the first iteration in phase II as round 0.

Definition 3. Let $\boldsymbol{x}_i^*$ be an optimal bundle of some agent $i \in \mathcal{A}$. For any round $r \geq 1$ in phase II, the set $\mathcal{L}_{i,r} \subset \boldsymbol{x}_i^*$ of *lost* items is the set of all items $j \in \boldsymbol{x}_i^*$ assigned to other agents $i' \neq i$ in that round.

Definition 4. Let $\boldsymbol{x}_i^*$ be an optimal bundle of some agent $i \in \mathcal{A}$ and $\boldsymbol{x}_i^{\mathrm{II}} = \{h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}\}$ be her bundle in phase II. The set $\bar{\boldsymbol{x}}_{i,r}^*$ of *optimal and attainable* items is defined as $\bar{\boldsymbol{x}}_{i,0}^* := \boldsymbol{x}_i^* \setminus \bigcup_{i' \in \mathcal{A}} \boldsymbol{x}_{i'}^{\mathrm{I}}$ in round 0 and as $\bar{\boldsymbol{x}}_{i,r}^* := \bar{\boldsymbol{x}}_{i,r-1}^* \setminus (\mathcal{L}_{i,r} \cup \{h_i^{r-1}\}$ in round $r \in [1, \tau_i^{\mathrm{II}}]$.

We denote their sizes by $\ell_{i,r} := |\mathcal{L}_{i,r}|$ and $\bar{\tau}_{i,r}^* := |\bar{\boldsymbol{x}}_{i,r}^*|$, respectively. First, we give a lower bound on the valuations of optimal and attainable items.

i: Would it be ‘dirty’ to include notation [redacted]n the [redacted]efini- [redacted]ions?

Lemma 3. *For each agent $i \in \mathcal{A}$ and her bundle $\boldsymbol{x}_i^{\mathrm{II}} = \{h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}\}$, it holds in all rounds $r = 2, \dots, \tau_i^{\mathrm{II}}$ of phase II that*

$$v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^{r-1}) \geq v_i(\bar{\boldsymbol{x}}_{i,1}^*) - \ell_{i,2} \cdot v_i(h_i^1) - \sum_{r'=2}^{r-1} \ell_{i,r'+1} \cdot v_i(h_i^{r'} \mid h_i^1, \dots, h_i^{r'-1}) - v_i(h_i^1, \dots, h_i^{r-1}).$$

Proof. We prove the lemma by induction on the number r of rounds. In the beginning of the base case $r = 2$, agent i has already been assigned item h_i^1. For each of the optimal and attainable items $j \in \bar{\boldsymbol{x}}_{i,1}^*$ in round 1, the marginal valuation $v_i(j \mid \emptyset)$ over the empty set was at most $v_i(h_i^1 \mid \emptyset)$, as otherwise item h_i^1 would not have been assigned first. The marginal valuation $v_i(j \mid h_i^1)$ over $\{h_i^1\}$ is upper-bounded by $v_i(h_i^1 \mid \emptyset)$, too, due to the submodularity of valuations. During round 2, a further $\ell_{i,2}$ of these items j are assigned to other agents, and item h_i^2 is assigned to agent i. We can bound the marginal valuation of the remaining optimal and attainable items in round 2 in the following way:

Case 1 — $h_i^1 \in \bar{\boldsymbol{x}}_{i,1}^*$: It holds $v_i(\bar{\boldsymbol{x}}_{i,2}^* \mid h_i^1) = v_i(\bar{\boldsymbol{x}}_{i,2}^* \cup \{h_i^1\}) - v_i(h_i^1) = v_i(\bar{\boldsymbol{x}}_{i,1}^* \setminus \mathcal{L}_{i,2}) - v_i(h_i^1)$.

Case 2 — $h_i^1 \notin \bar{\boldsymbol{x}}_{i,1}^*$: Due to the monotonicity of the valuation functions, it holds $v_i(\bar{\boldsymbol{x}}_{i,2}^* \cup \{h_i^1\}) \geq v_i(\bar{\boldsymbol{x}}_{i,2}^*)$ and, therefore, $v_i(\bar{\boldsymbol{x}}_{i,2}^* \mid h_i^1) \geq v_i(\bar{\boldsymbol{x}}_{i,2}^*) - v_i(h_i^1) = v_i(\bar{\boldsymbol{x}}_{i,1}^* \setminus \mathcal{L}_{i,2}) - v_i(h_i^1)$.

In both cases, the base case is proven because

$$\begin{aligned} v_i(\bar{\boldsymbol{x}}_{i,2}^* \mid h_i^1) &\geq v_i(\bar{\boldsymbol{x}}_{i,1}^* \setminus \mathcal{L}_{i,2}) - v_i(h_i^1) && (13)\\ &\geq v_i(\bar{\boldsymbol{x}}_{i,1}^*) - v_i(\mathcal{L}_{i,2}) - v_i(h_i^1) && (14)\\ &\geq v_i(\bar{\boldsymbol{x}}_{i,1}^*) - \ell_{i,2} \cdot v_i(h_i^1) - v_i(h_i^1), && (15) \end{aligned}$$

where the second inequality is shown easily using an alternative definition of submodularity $(v_i(\mathcal{S}_1 \cup \mathcal{S}_2) + v_i(\mathcal{S}_1 \cap \mathcal{S}_2) \leq v_i(\mathcal{S}_1) + v_i(\mathcal{S}_2)$ with $\mathcal{S}_1 = \bar{\boldsymbol{x}}_{i,1}^* \setminus \mathcal{L}_{i,2}$ and $\mathcal{S}_2 = \mathcal{L}_{i,2}$ [20]), and the third inequality is due all $\ell_{i,2}$ items j in set $\mathcal{L}_{i,2}$ not being assigned in round 1 although attainable, implying $v_i(j) \le v_i(h_i^1)$.

For the induction hypothesis, we assume that the lemma holds true for all rounds up to some r. In the induction step $r \to r+1$, we differentiate the same two cases again:

Case 1—$h_i^r \in \bar{\boldsymbol{x}}_{i,r}^*$: Again we exploit the submodularity of the valuation functions to obtain a lower bound on the marginal valuation of $\bar{\boldsymbol{x}}_{i,r+1}^*$.

$$\begin{aligned}
v_i(\bar{\boldsymbol{x}}_{i,r+1}^* \mid h_i^1, \dots, h_i^r) &= v_i(\bar{\boldsymbol{x}}_{i,r+1}^* \cup \{h_i^r\} \mid h_i^1, \dots, h_i^{r-1}) - v_i(h_i^r \mid h_i^1, \dots, h_i^{r-1}) && (16)\\
&= v_i(\bar{\boldsymbol{x}}_{i,r}^* \setminus \mathcal{L}_{i,r+1} \mid h_i^1, \dots, h_i^{r-1}) - v_i(h_i^r \mid h_i^1, \dots, h_i^{r-1}) && (17)\\
&\ge v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^{r-1}) - v_i(h_i^r \mid h_i^1, \dots, h_i^{r-1}) && (18)\\
&\qquad - v_i(\mathcal{L}_{i,r+1} \mid h_i^1, \dots, h_i^{r-1})
\end{aligned}$$

Case 2—$h_i^r \notin \bar{\boldsymbol{x}}_{i,r}^*$: At first, we use the monotonicity of the valuation functions to get the inequality

$$\begin{aligned}
v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^r) &= v_i(\bar{\boldsymbol{x}}_{i,r}^* \cup \{h_i^1, \dots, h_i^r\}) - v_i(h_i^1, \dots, h_i^r) && (19)\\
&\ge v_i(\bar{\boldsymbol{x}}_{i,r}^* \cup \{h_i^1, \dots, h_i^{r-1}\}) - v_i(h_i^1, \dots, h_i^r) && (20)\\
&= \left(v_i(\bar{\boldsymbol{x}}_{i,r}^* \cup \{h_i^1, \dots, h_i^{r-1}\}) - v_i(h_i^1, \dots, h_i^{r-1})\right) && (21)\\
&\qquad - \left(v_i(h_i^1, \dots, h_i^r) - v_i(h_i^1, \dots, h_i^{r-1})\right)\\
&= v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^{r-1}) - v_i(h_i^r \mid h_i^1, \dots, h_i^{r-1}). && (22)
\end{aligned}$$

Together with the submodularity of valuation, we obtain the same lower bound again:

$$\begin{aligned}
v_i(\bar{\boldsymbol{x}}_{i,r+1}^* \mid h_i^1, \dots, h_i^r) &= v_i(\bar{\boldsymbol{x}}_{i,r}^* \setminus \mathcal{L}_{i,r+1} \mid h_i^1, \dots, h_i^r) && (23)\\
&\ge v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^r) - v_i(\mathcal{L}_{i,r+1} \mid h_i^1, \dots, h_i^r) && (24)\\
&\ge v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^r) - v_i(\mathcal{L}_{i,r+1} \mid h_i^1, \dots, h_i^{r-1}) && (25)\\
&\ge v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^{r-1}) - v_i(h_i^r \mid h_i^1, \dots, h_i^{r-1}) && (26)\\
&\qquad - v_i(\mathcal{L}_{i,r+1} \mid h_i^1, \dots, h_i^{r-1})
\end{aligned}$$

In both cases, we can replace $v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^{r-1})$ by the induction hypothesis and $v_i(\mathcal{L}_{i,r+1} \mid h_i^1, \dots, h_i^{r-1})$ by $\ell_{i,r+1} \cdot v_i(h_i^r \mid h_i^1, \dots, h_i^{r-1})$ to prove the lemma. For a detailed calculation we refer to Garg, Kulkarni and Kulkarni [15, p. 14]. ◻

The lemma can be used to find a lower bound on the marginal valuation of the items assigned in each round r.

Corollary 1. *From lemma 3 follows*

$$v_i(h_i^r \mid h_i^1, \dots, h_i^{r-1}) \ge \Big(v_i(\bar{\boldsymbol{x}}_{i,1}^*) - \ell_{i,2} \cdot v_i(h_i^1) - \sum_{r'=2}^{r-1} \ell_{i,r'+1} \cdot v_i(h_i^{r'} \mid h_i^1, \dots, h_i^{r'-1}) - v_i(h_i^1, \dots, h_i^{r-1})\Big) \Big/ \bar{\tau}_{i,r}^*.$$

Proof. Remember that valuation functions are monotonic if $v_i(\mathcal{S}_1) \le v_i(\mathcal{S}_2)$ holds for all sets $\mathcal{S}_1 \subset \mathcal{S}_2 \subset \mathcal{G}$. Induction shows that there must be an item $j \in \mathcal{S}_2$ with valuation $v_i(j) \ge v_i(\mathcal{S}_2)/|\mathcal{S}_2|$, otherwise it would hold $v_i(\emptyset) > 0$. Applied to lemma 3, this means that there must be an item $j \in \bar{\boldsymbol{x}}_{i,r}^*$ with a marginal valuation of at least $v_i(\bar{\boldsymbol{x}}_{i,r}^* \mid h_i^1, \dots, h_i^{r-1})/\bar{\tau}_{i,r}^*$. As item h_i^r was the one to be assigned, its marginal valuation cannot be smaller. ◻

This, finally, enables us to give a lower bound on the valuation of the bundles $\boldsymbol{x}_i^{\mathrm{II}}$.

Lemma 4. *For each agent $i \in \mathcal{A}$ and her bundle $\boldsymbol{x}_i^{\mathrm{II}} = \{h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}\}$, it holds*

$$v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) \geq v_i(\bar{\boldsymbol{x}}_{i,1}^*)/n.$$

Proof. In each round $r = 1, \dots, \tau_i^{\mathrm{II}}$, $\ell_{i,r}$ optimal and attainable items of agent i are assigned to other agents. As there are n agents in total, $n-1$ is an upper bound on $\ell_{i,r}$. Furthermore, after τ_i^{II} rounds, the number $\bar{\tau}_{i,\tau_i^{\mathrm{II}}}^*$ of optimal and attainable items is at most $n - 1 \leq n$ elsewise agent i would have been assigned yet another item. Together with corollary 1, this proves the lemma:

$$\begin{aligned}
v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) &= v_i(h_i^{\tau_i^{\mathrm{II}}} \mid h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) + v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1}) && (27)\\
&\geq \Big(v_i(\bar{\boldsymbol{x}}_{i,1}^*) - \ell_{i,2} \cdot v_i(h_i^1) - \sum_{r'=2}^{\tau_i^{\mathrm{II}}-1} \ell_{i,r'+1} \cdot v_i(h_i^{r'} \mid h_i^1, \dots, h_i^{r'-1}) && (28)\\
&\qquad - v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1})\Big) \Big/ \bar{\tau}_{i,r}^* + v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1})\\
&\geq \Big(v_i(\bar{\boldsymbol{x}}_{i,1}^*) - (n-1) v_i(h_i^1) - \sum_{r'=2}^{\tau_i^{\mathrm{II}}-1} (n-1) v_i(h_i^{r'} \mid h_i^1, \dots, h_i^{r'-1}) && (29)\\
&\qquad - v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1})\Big) \Big/ n + v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1})\\
&\geq (v_i(\bar{\boldsymbol{x}}_{i,1}^*) - (n-1) v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1}) - v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1}))/n && (30)\\
&\qquad + v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}-1})\\
&= v_i(\bar{\boldsymbol{x}}_{i,1}^*)/n && (31)
\end{aligned}$$

□

After having obtained a lower bound on the valuation of items assigned in phase II, we need a lower bound for phase III as well. Therefor we introduce a third type of item set.

Definition 5. Let $\boldsymbol{x}_i^* = \{g_i^1, \dots, h_i^{\tau_i^*}\}$ be an optimal bundle of some agent $i \in \mathcal{A}$. The set $\mathcal{G}_i^+$ of *overly good* items is defined as $\mathcal{G}_i^+ := \{\, j \in \mathcal{G} \mid v_i(j) \geq v_i(g_i^1) \,\}$.

Lemma 5. *In phase III, there exists a matching such that each agent $i \in \mathcal{A}$ is matched to one of her overly good items in the set $\bigcup_{i' \in \mathcal{A}} \boldsymbol{x}_{i'}^{\mathrm{I}}$ of released items.*

Proof. If all items were matched in phase I, i. e., $\bigcup_{i' \in \mathcal{A}} \boldsymbol{x}_{i'}^{\mathrm{I}} = \mathcal{G}$, then all optimal items are released in phase III and each agent can be matched to one; the lemma is proven immediately. If not, imagine for some t that only the items assigned in the first t rounds of phase I were released. Now choose some matching $\mathcal{M}_t$ with the following properties:

1. If for an agent i all overly good items were amongst the released items, she gets matched with an overly good item $j \in \mathcal{G}_i^+$.
2. The number of agents matched with one of their overly good items is maximal amongst all matchings fulfilling property 1.

Property 1 is always satisfiable as the union of k many sets $\mathcal{G}_i^+$ contains k different items g_i^1, which can be matched with agents i. Property 2 leads to all agents being matched with an overly good item for $t = \lceil \log_2 n \rceil + 1$, i. e. the number of rounds in phase I, whence the lemma follows. To prove this, we denote by $\mathcal{A}_t^-$ the set of agents who are *not* matched with one of their overly good items, and show by induction on t that it holds $|\mathcal{A}_t^-| \leq n/2^t$.

In the base case $t = 1$, none of the items are assigned initially. Denote by α the number of agents who were not assigned an overly good item in the first round of phase I. If $\alpha \leq n/2$, then a matching $\mathcal{M}_1$ obviously exists and the base case is immediately proven. Otherwise, all items from at least α many sets $\mathcal{G}_i^+$ got assigned to someone. Again: Each set $\mathcal{G}_i^+$ is the only one containing the item g_i^1, so the union of these sets contains at least α items which can be matched with at least α agents upon release. This then leaves at most $n - \alpha < n/2$ agents not matched with an overly good item.

For the induction hypothesis, we assume that the statement holds true for all rounds up to some t. In the induction step $t \to t+1$, by property 1, there is at least one unassigned item in each set $\mathcal{G}_{i'}^+$ for all agents $i' \in \mathcal{A}_t^-$ at the start of round $t+1$. Analogously to the base case, for at least half of those agents i' these unassigned items will be assigned to them or someone else and it can be argued accordingly. By the induction hypothesis, it holds $|\mathcal{A}_{t+1}^-| \leq |\mathcal{A}_t^-|/2 \leq (n/2^t)/2 = n/2^{t+1}$. ∎

This allows us to calculate an approximation factor for RepReMatch by comparing its output with an optimal allocation $\boldsymbol{x}^*$.

Theorem 2. *RepReMatch is* $(2n(\log_2 n + 3))$*-approximative.*

Proof. By lemma 5, we can assign each agent i an overly good item $j_i^+ \in \mathcal{G}_i^+$ in the beginning of phase III. RepReMatch maximises the logarithmic Nash social welfare, so

$$\log \mathrm{NSW}(\boldsymbol{x}^{\mathrm{III}}) \geq \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log v_i(j_i^+, h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) \tag{32}$$

is a lower bound on the logarithmic NSW after the first matching in phase III, whereby $\boldsymbol{x}_i^{\mathrm{II}} = \{h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}\}$ is the bundle of agent i from phase II.

Item j_i^+ was released in phase III, which means it was assigned in phase I, implying $j_i^+ \in \boldsymbol{x}_i^* \setminus \bar{\boldsymbol{x}}_{i,0}^*$ and, subsequently, $j_i^+ \in (\boldsymbol{x}_i^* \setminus \bar{\boldsymbol{x}}_{i,0}^*) \cup \mathcal{L}_{i,1}$. Phase I runs for at most $\lceil \log_2 n \rceil + 1$ rounds, and at most n items are assigned in each iteration. Therefore, at most $n(\log_2 n + 2)$ optimal items are assigned in that phase, i. e., $|\boldsymbol{x}_i^* \setminus \bar{\boldsymbol{x}}_{i,0}^*| \leq n(\log_2 n + 2)$. Furthermore, it holds $n \geq \ell_{i,1} = |\mathcal{L}_{i,1}|$ as in lemma 4. Together with the monotonicity of the valuation functions, this yields

i: Error in paper? see also lemma 5

$$v_i(j_i^+, h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) \geq v_i(j_i^+) \geq \frac{v_i((\boldsymbol{x}_i^* \setminus \bar{\boldsymbol{x}}_{i,0}^*) \cup \mathcal{L}_{i,1})}{n(\log_2 n + 3)} \tag{33}$$

as lower bound on the valuations of bundles. Moreover, lemma 2 and the monotonicity of the valuation functions functions yield

$$v_i(j_i^+, h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) \geq v_i(h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) \geq \frac{v_i(\bar{\boldsymbol{x}}_{i,1}^*)}{n} \geq \frac{v_i(\bar{\boldsymbol{x}}_{i,1}^*)}{n(\log_2 n + 3)} = \frac{v_i(\bar{\boldsymbol{x}}_{i,0}^* \setminus \mathcal{L}_{i,1})}{n(\log_2 n + 3)} \tag{34}$$

as yet another lower bound. The mean of eqs. (33) to (34) and the monotonicity of the valuation functions give the concise lower bound

$$\begin{aligned}
v_i(j_i^+, h_i^1, \dots, h_i^{\tau_i^{\mathrm{II}}}) &\geq \frac{1}{2}\left(\frac{v_i((\boldsymbol{x}_i^* \setminus \bar{\boldsymbol{x}}_{i,0}^*) \cup \mathcal{L}_{i,1})}{n(\log_2 n + 3)} + \frac{v_i(\bar{\boldsymbol{x}}_{i,0}^* \setminus \mathcal{L}_{i,1})}{n(\log_2 n + 3)}\right) && (35)\\
&\geq \frac{1}{2} \cdot \frac{v_i(((\boldsymbol{x}_i^* \setminus \bar{\boldsymbol{x}}_{i,0}^*) \cup \mathcal{L}_{i,1}) \cup (\bar{\boldsymbol{x}}_{i,0}^* \setminus \mathcal{L}_{i,1}))}{n(\log_2 n + 3)} && (36)\\
&= \frac{v_i(\boldsymbol{x}_i^*)}{2n(\log_2 n + 3)}. && (37)
\end{aligned}$$

We can insert this lower bound into eq. (32) and prove the theorem thereby:

$$\log \mathrm{NSW}(\boldsymbol{x}^{\mathrm{III}}) \geq \frac{1}{\sum_{i \in \mathcal{A}} \eta_i} \cdot \sum_{i \in \mathcal{A}} \eta_i \log\left(\frac{v_i(\boldsymbol{x}_i^*)}{2n(\log_2 n + 3)}\right) = \log\left(\frac{\mathrm{NSW}(\boldsymbol{x}^*)}{2n(\log_2 n + 3)}\right) \tag{38}$$

∎

overly good -> outstanding?

4 Hardness of Approximation

Garg, Kulkarni and Kulkarni [15, Sction 4] provide the following hardness result.

Theorem 3. *The submodular NSW is not approximable within a factor of $\frac{\mathrm{e}}{\mathrm{e}-1}$ in polynomial time unless $\mathcal{P} = \mathcal{NP}$, even when the agents have equal weights and valuation functions.*

i: Should it not be $\frac{\mathrm{e}}{\mathrm{e}-1} - \epsilon$?

Proof. Consider the related USW problem[1].

Problem 2. Given a set $\mathcal{G}$ of indivisible items and a set $\mathcal{A}$ of agents with monotonic, submodular valuation functions $v_i \colon \mathcal{P}(\mathcal{G}) \to \mathbb{R}$ for all agents $i \in \mathcal{A}$, the *symmetric submodular utilitarian social welfare problem* is to find an allocation $\boldsymbol{x}^*$ maximising the sum of valuations, that is,

$$\boldsymbol{x}^* \overset{!}{=} \underset{\boldsymbol{x} \in \boldsymbol{X}_{\mathcal{A}}(\mathcal{G})}{\arg\max} \{\mathrm{USW}(\boldsymbol{x})\} \quad \text{with } \mathrm{USW}(\boldsymbol{x}) := \sum_{i \in \mathcal{A}} v_i(\boldsymbol{x}_i)$$

where $\boldsymbol{X}_{\mathcal{A}}(\mathcal{G})$ is the set of all possible allocations of the items in $\mathcal{G}$ amongst n agents.

Note that this problem is identical to the symmetric submodular NSW except for the sum in the target function instead of a product. We will exploit this fact to calculate the NSW of instances for USW. Khot et al. [20] supply a polynomial-time reduction of USW from the following problem:

Problem 3. Given a graph $G = (V, E)$ and a constant $c \leq 1$, the *c-Gap-Max-3-Colouring problem* is to decide whether, for any 3-colouring of graph G which maximises the number of edges with different coloured endpoints, the number of such edges is $|E|$ (*Yes instance*) or $c|E|$ and below (*No instance*).

Proposition 1. *There exists a constant $c \leq 1$ such that the c-Gap-Max-3-Colouring problem is $\mathcal{NP}$-hard.*

Reducing an instance of the c-Gap-Max-3-Colouring problem yields an instance of the symmetric submodular USW problem with identical valuation functions. Its properties are as follows:

Yes instance The USW is nC because every agent values her bundle at n, whereby C is a constant depending on the input graph. The NSW of the instance would be C.

No instance The USW is $\frac{\mathrm{e}-1}{\mathrm{e}} nC$. Applying the inequality of arithmetic and geometric means, i. e., $(x_1 + \dots + x_n)/n \geq \sqrt[n]{x_1 \cdots x_n}$ for all nonnegative numbers $x_1, \dots, x_n \in \mathbb{R}_0^+$, reveals that the NSW of the instance is at most $\frac{\mathrm{e}-1}{\mathrm{e}} C$.

Thereout follows that the submodular NSW problem, even when symmetric and with identical valuation functions, cannot be approximated within a factor better than $\frac{\mathrm{e}}{\mathrm{e}-1}$; otherwise one could decide the c-Gap-Max-3-Colouring problem in polynomial time by checking whether the corresponding NSW instance has a value above $\frac{\mathrm{e}-1}{\mathrm{e}} C$. ☐

For a constant number of agents, Garg, Kulkarni and Kulkarni [15, Section 5.1] describe a family $(A_\epsilon)_{\epsilon > 0}$ of algorithms for the asymmetric submodular NSW problem where each algorithm A_ϵ achieves an approximation factor of $\frac{\mathrm{e}}{\mathrm{e}-1} + \epsilon$.

[1]Garg, Kulkarni and Kulkarni (and many others) call problem 2 the 'Allocation problem'. We changed the name to match the naming scheme of the NSW problem and to avoid confusion, as both problems are about finding allocations.

5 Conclusion and Outlook

In this seminar report, we presented two polynomial-time algorithms for the asymmetric Nash social welfare problem (NSW), based on a paper by Garg, Kulkarni and Kulkarni [15]. The asymmetric NSW asks to find an allocation of unsharable and indivisible items amongst agents such that the weighted geometric mean of their valuations is maximised. The novelty lies in both algorithms having an approximation factor dependant on the number n of agents but not on the number m of items.

The first algorithm, SMatch, finds a $2n$-approximative allocation if the valuation functions are additive. It does so by repeatedly matching agents with items within a weighted bipartite graph. For the very first matching, the edge weights incorporate an estimation of the valuation of future items. The output allocation is envy-free up to one item.

The second algorithm, RepReMatch, finds a $2n(\log_2 n + 3)$-approximative allocation if the valuation functions are submodular. In phase I, a set of high-value items is determined through repeated matchings and then put away. In phase II, agents are repeatedly matched with the remaining items. In phase III, the high-value items are finally assigned to the agents.

Lastly, it was shown that any polynomial-time algorithm for the submodular NSW must have an approximation factor of at least $\frac{\mathrm{e}}{\mathrm{e}-1}$ unless $\mathcal{P} = \mathcal{NP}$. This holds even when the problem is symmetric and the valuation functions equal.

Garg, Kulkarni and Kulkarni [15] conjecture that SMatch has a better approximation factor for the symmetric additive NSW though neither were they able to prove a better factor nor could they prove the tightness of their given analysis. Additionally, they forbore to prove the tightness of the analysis of RepReMatch, and we suspect there to be quite some constants to be saved — admittedly, the benefits of a more thorough analysis are questionable. To identify more general open questions, we need to take a look at new publications since 2020, the year of the publication of the paper by Garg, Kulkarni and Kulkarni.

- Rado valuations are a special class of submodular functions and stem from a generalisation of OXS valuations. Garg, Kulkarni and Kulkarni emphatically mentioned the lack of research into the OXS NSW. Garg, Husic and Vegh [14] developed an algorithm for the Rado NSW, which is independent from both the number of items and the number of agents. For the symmetric Rado NSW, their algorithm achieves an approximation factor of $256\mathrm{e}^{3/\mathrm{e}} \approx 772$. For the asymmetric Rado NSW, the approximation factor is $256\gamma^3$ with $\gamma := \max_{i\in\mathcal{A}}\{\eta_i\}/\min_{i\in\mathcal{A}}\{\eta_i\}$. Interestingly, the algorithm is 16γ-approximative in case of the asymmetric additive NSW, so it outperforms SMatch in many instances. Remarkably enough, the algorithm is divided into five phases, and the first phase serves to determine a set of high-value items.
- Li and Vondrák [21] introduced a randomised, 380-approximative algorithm for the symmetric submodular NSW. Later, Garg et al. [16] devised a family of deterministic algorithms for every $\epsilon > 0$. They are $(4 + \epsilon)$-approximative in the symmetric case and $\mathrm{e}(n \cdot \max_{i\in\mathcal{A}}\{\eta_i\} + 2 + \epsilon)$-approximative in the asymmetric one.
- XOS functions are a superclass of submodular functions, for which the approximation factor of RepReMatch is not independent of the number of items anymore [15, Section 6.2]. Barman et al. [4] used both RepReMatch and the discrete moving-knife method to get an $\mathcal{O}(n^{53/54})$-approximative algorithm for the symmetric XOS NSW. However, the algorithm uses demand and XOS queries, whereas RepReMatch needs only the weaker value queries.
- CASC (also known as SPLC) functions are a superclass of additive functions, which can

model diminishing returns. Chaudhury et al. [7] came up with an $\mathrm{e}^{1/\mathrm{e}}$-approximative algorithm for the symmetric CASC NSW.

- Garg et al. [17] offer (fully) polynomial-time approximation schemes and even an optimal algorithm for some types of the symmetric and asymmetric additive NSW.

Besides efficiency, fairness is also a property towards which algorithm can be designed, although SMatch and RepReMatch advance little in that regard. Especially RepReMatch with no fairness guarantees is unsatisfactory. Research is made complex because of the myriad of notions of fairness, though some sort of envy-freeness is commonly used. A selection of new developments:

- The algorithm of Chaudhury et al. [7] computes an allocation which is $1/2$-EF1 and approximately Pareto-optimal.
- Best of Both Worlds describes an approach where randomisation for fairness in expectation and a relaxation of fairness criteria are combined. Hoefer, Schmalhofer and Varricchio [18] use it to provide fair allocations for the asymmetric additive NSW and the symmetric XOS NSW.
- Feldman, Mauras and Ponitka [12] show a $\frac{1}{\alpha+1}$-approximative algorithm computing allocations which are α-EFX for the symmetric additive NSW with $\alpha \in [0, 1]$. They also show a $\frac{1}{\alpha+1}$-approximative algorithm computing allocations which are α-EFX for the symmetric subadditive NSW with $\alpha \in [0, \frac{1}{2}]$. Subadditive functions are a superclass for XOS functions.

As seen, the research on the NSW gained pace in the last years, and some points raised by Garg, Kulkarni and Kulkarni have been addressed. Apart from devising algorithms with other valuation functions or restrictions close to reality, it could be rewarding to look at more basic cases. For example, Garg, Hoefer and Mehlhorn [13] provide a hardness of $\sqrt{8/7} > 1.069$ for the symmetric additive NSW, but the best known algorithm is that of Barman, Krishnamurthy and Vaish [3] with an approximation factor of $\mathrm{e}^{1/\mathrm{e}} \approx 1.45$.

References

[1] Arash Asadpour and Amin Saberi. 'An Approximation Algorithm for Max-Min Fair Allocation of Indivisible Goods'. In: *SIAM Journal on Computing* 39.7 (2010), pp. 2970–2989. DOI: 10.1137/080723491 (visited on 10/06/2023).

[2] Tayebeh Bahreini, Hossein Badri and Daniel Grosu. 'An Envy-Free Auction Mechanism for Resource Allocation in Edge Computing Systems'. In: *SEC 2018*. 2018 IEEE/ACM Symposium on Edge Computing (Seattle, WA, USA, 25th–27th Oct. 2018). Ed. by Patrick Kellenberger. The Institute of Electrical and Electronics Engineers, Inc., 2018, pp. 313–322. ISBN: 978-1-5386-9446-6. DOI: 10.1109/SEC.2018.00030. URL: https://www.researchgate.net/publication/329561811_An_Envy-Free_Auction_Mechanism_for_Resource_Allocation_in_Edge_Computing_Systems (visited on 08/06/2023).

[3] Siddharth Barman, Sanath Kumar Krishnamurthy and Rohit Vaish. *Finding Fair and Efficient Allocations*. 11th May 2018. arXiv: 1707.04731 [cs.GT] (visited on 11/06/2023).

[4] Siddharth Barman et al. *Sublinear Approximation Algorithm for Nash Social Welfare with XOS Valuations*. 15th July 2022. arXiv: 2110.00767 [cs.GT] (visited on 11/06/2023).

[5] Ivona Bezáková and Varsha Dani. 'Allocating Indivisible Goods'. In: *ACM SIGecom Exchanges* 5.3 (1st Apr. 2005), pp. 11–18. DOI: 10.1145/1120680.1120683. URL: https://newtraell.cs.uchicago.edu/files/tr_authentic/TR-2004-10.pdf (visited on 11/06/2023).

[6] Ioannis Caragiannis et al. 'The Unreasonable Fairness of Maximum Nash Welfare'. In: *EC '16*. 17th ACM Conference on Economics and Computation (Maastricht, The Netherlands, 24th–28th July 2016). Ed. by Vincent Conitzer, Dirk Bergemann and Yiling Chen. Association for Computing Machinery, 2016, pp. 305–322. ISBN: 978-1-4503-3936-0. DOI: 10.1145/2940716. URL: https://eprints.gla.ac.uk/123283 (visited on 10/06/2023).

[7] Bhaskar Ray Chaudhury et al. 'Fair Division of Indivisible Goodsfor a Class of Concave Valuations'. In: *Journal of Artificial Intelligence Researc* 74 (May 2022), pp. 111–142. DOI: 10.1613/jair.1.12911 (visited on 12/06/2023).

[8] Yann Chevaleyre et al. 'Issues in Multiagent Resource Allocation'. In: *Informatica* 30.1 (Jan. 2006), pp. 3–31. URL: https://www.informatica.si/index.php/informatica/article/view/70/62 (visited on 07/06/2023).

[9] Richard Cole and Vasilis GKatzelis. 'Approximating the Nash Social Welfare with Indivisible Items'. In: *SIAM Journal on Computing* 47.3 (2018), pp. 1211–1236. DOI: 10.1137/15M1053682 (visited on 10/06/2023).

[10] Dagmawi Mulugeta Degefu et al. 'Bankruptcy to Surplus: Sharing Transboundary River Basin's Water under Scarcity'. In: *Water Resources Management* 32 (8 1st June 2018), pp. 2735–2751. ISSN: 1573-1650. DOI: 10.1007/s11269-018-1955-z (visited on 08/06/2023).

[11] Schahram Dustdar, ed. *NSF Workshop Report on Grand Challenges in Edge Computing*. NSF Edge Computing Workshop (Washington, DC, USA, 26th Oct. 2016). National Science Foundation. 15th Nov. 2017. 17 pp. URL: https://dsg.tuwien.ac.at/team/sd/papers/Bericht_NSF_S_Dustdar.pdf (visited on 08/06/2023).

[12] Michal Feldman, Simon Mauras and Tomasz Ponitka. *On Optimal Tradeoffs between EFX and Nash Welfare*. 19th Feb. 2023. arXiv: 2302.09633 [cs.GT] (visited on 14/06/2023).

[13] Jugal Garg, Martin Hoefer and Kurt Mehlhorn. *Satiation in Fisher Markets and Approximation of Nash Social Welfare*. 31st July 2019. arXiv: 1707.04428 [cs.DS] (visited on 11/06/2023).

[14] Jugal Garg, Edin Husic and Laszlo A. Vegh. *Approximating Nash Social Welfare under Rado Valuations*. 30th Sept. 2020. arXiv: 2009.14793 [cs.GT] (visited on 11/06/2023).

[15] Jugal Garg, Pooja Kulkarni and Rucha Kulkarni. *Approximating Nash Social Welfare under Submodular Valuations through (Un)Matchings*. 28th Dec. 2019. arXiv: 1912.12541v1 [cs.GT].

[16] Jugal Garg et al. *Approximating Nash Social Welfare by Matching and Local Search*. 29th Mar. 2023. arXiv: 2211.03883 [cs.GT] (visited on 11/06/2023). To appear in STOC '23.

[17] Jugal Garg et al. *Tractable Fragments of the Maximum Nash Welfare Problem*. 28th Apr. 2022. arXiv: 2112.10199 [cs.GT] (visited on 14/06/2023).

[18] Martin Hoefer, Marco Schmalhofer and Giovanna Varricchio. *Best of Both Worlds: Agents with Entitlements*. arXiv: 2209.03908 [cs.GT] (visited on 12/06/2023).

[19] Subhash Khot and Ashok Kumar Ponnuswami. 'Approximation Algorithms for the Max-Min Allocation Problem'. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007 (Princeton, NJ, USA, 20th–22nd Aug. 2007). Ed. by Moses Charikar et al. Berlin, Heidelberg: Springer, 2007, pp. 204–217. ISBN: 978-3-540-74207-4. DOI: 10.1007/978-3-540-74208-1_15 (visited on 10/06/2023).

[20] Subhash Khot et al. 'Inapproximability Results for Combinatorial Auctions with Submodular Utility Functions'. In: *Algorithmica* 52 (1 13th Oct. 2007), pp. 3–18. ISSN: 0178-4617. DOI: 10.1007/s00453-007-9105-7 (visited on 05/06/2023).

[21] Wenzheng Li and Jan Vondrák. *A constant-factor approximation algorithm for Nash Social Welfare with submodular valuations*. 16th Nov. 2021. arXiv: 2103.10536 [cs.GT] (visited on 11/06/2023).

[22] Trung Thanh Nguyen and Jörg Rothe. 'Minimizing envy and maximizing average Nash social welfare in the allocation of indivisible goods'. In: *Discrete Applied Mathematics* 179 (2014), pp. 54–68. ISSN: 0166-218X. DOI: 10.1016/j.dam.2014.09.010 (visited on 11/06/2023).

[23] Zoya Svitkina and Lisa Fleischer. *Submodular Approximation: Sampling-based Algorithms and Lower Bounds*. 31st May 2010. arXiv: 0805.1071 [cs.DS] (visited on 01/06/2023).

[24] Jan Vondrák. 'Optimal Approximation for the Submodular Welfare Problem in the Value Oracle Model'. In: *STOC '08*. 40th ACM Symposium on Theory of Computing (Victoria, British Columbia, Canada, 17th–20th May 2008). Ed. by editor. Association for Computing Machinery, 2008, pp. 67–74. ISBN: 978-1-60558-047-0. DOI: 10.1145/3313276.3316304. URL: https://theory.stanford.edu/~jvondrak/data/submod-value.pdf (visited on 10/06/2023).