Algorithms and Complexity Group
Institute for Computer Science
Goethe University Frankfurt

Summer Semester 2023

Seminar Approximation Algorithms

ANSWuSVþ(U)M

Zeno Adrian Weil

21st May 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

- problem introduction, motivation, applications
- formal problem definition
- short literature review: What is known, what not? New findings?
- content & structure of paper

Definition 1. Let $\mathcal{G} := \{1, \dots, m\}$ be a set of indivisible *items* and $\mathcal{A} := \{1, \dots, n\}$ be a set of *agents*. An *allocation* is a tuple $\boldsymbol{x} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \in \mathcal{P}(G)^n$ such that each item is element of exactly one set $\boldsymbol{x}_i$, 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 'allocated'?

$\vdots$

Definition 2. Given a set $\mathcal{G}$ of items and a set $\mathcal{A}$ of agents with *valuations* $v_i : \mathcal{P}(\mathcal{G}) \to \mathbb{R}$ and *agent weights* η_i for all agents $i \in \mathcal{A}$, the *Nash Social Welfare problem* (NSW) is to find an allocation maximising the weighted geometric mean of valuations, that is

less strict def of valuations; restriction for our case later on

$$\operatorname*{arg\,max}_{\boldsymbol{x} \in \Pi_n(\mathcal{G})} \left\{ \left(\prod_{i \in \mathcal{A}} v_i(\boldsymbol{x}_i)^{\eta_i} \right)^{1/\sum_{i \in \mathcal{A}} \eta_i} \right\}$$

where $\Pi_n(\mathcal{G})$ is the set of all possible allocations of the items in $\mathcal{G}$ amongst n agents. The problem is called *symmetric* if all agent weights η_i are equal, and *asymmetric* otherwise.

agents without items assigned have valuation zero $\to$ prevent

$\vdots$

In a slight abuse of notation, we omit curly braces delimiting a set in the arguments of a valuation function, so for example we write $v(j_1, j_2, \dots)$ to denote $v(\{j_1, j_2, \dots\})$.

$\vdots$

definition of approximation factor [def environment or in-text?]

$\vdots$

Garg, Kulkarni and Kulkarni [1] consider five different types of non-negative monotonically non-decreasing valuation functions of which we are going to consider only the following two due to space constraints:

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 $v_i(\mathcal{S}) = \sum_{j \in \mathcal{S}} v_i(j)$.

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

We use *additive NSW* and *submodular NSW* as shorthands for the Nash social welfare problems with additive and submodular valuation functions, respectively.

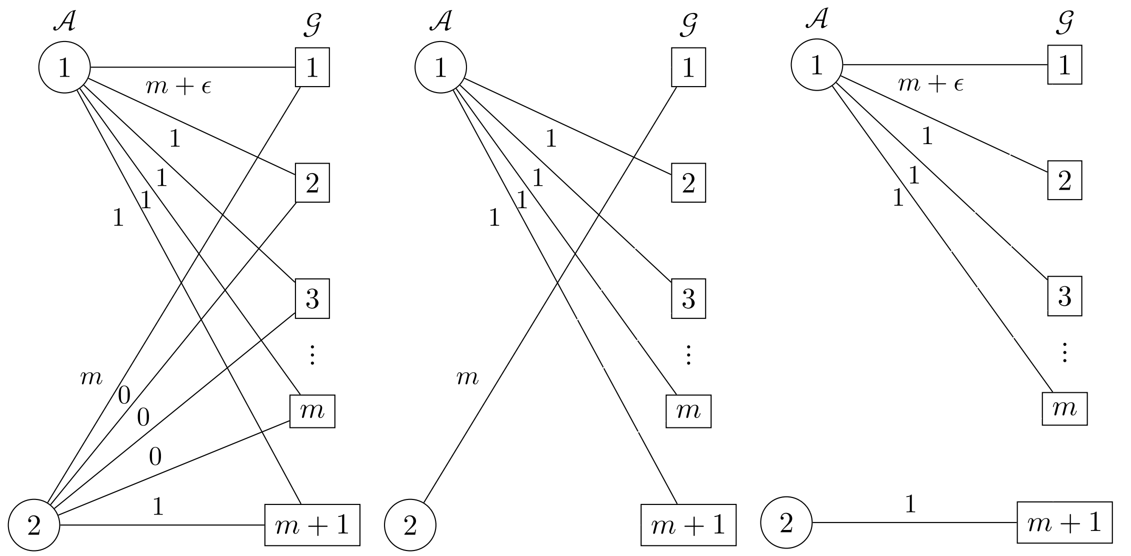
Morbi luctus, wisi viverra faucibus pretium, nibh est placerat odio, nec commodo wisi enim eget quam. Quisque libero justo, consectetuer a, feugiat vitae, porttitor eu, libero. Suspendisse sed mauris vitae elit sollicitudin malesuada. Maecenas ultricies eros sit amet ante. Ut venenatis velit. Maecenas sed mi eget dui varius euismod. Phasellus aliquet volutpat odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque sit amet pede ac sem eleifend consectetuer. Nullam elementum, urna vel imperdiet sodales, elit ipsum pharetra ligula, ac pretium ante justo a nulla. Curabitur tristique arcu eu metus. Vestibulum lectus. Proin mauris. Proin eu nunc eu urna hendrerit faucibus. Aliquam auctor, pede consequat laoreet varius, eros tellus scelerisque quam, pellentesque hendrerit ipsum dolor sed augue. Nulla nec lacus.

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi.

Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetuer eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor.

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo.

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.



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

might delete if document in two columns → unused edges in grey

Figure 1: Agent 1 values item 1 at $m+\epsilon$, and all other items at 1. Agent 2 values item 1 at m, item $m+1$ at 1, and all other items at 0. In an optimal allocation, item 1 would be assigned to agent 2 and all other items to agent 1, resulting in a NSW of $\sqrt{m \cdot m} = m$. A repeated maximum matching algorithm would greedily assign item 1 to agent 1 and item $m+1$ to agent 2 in the first round. Even if all remaining items were going to be assigned to agent 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. $|\mathcal{A}| = n = m = |\mathcal{G}|$, 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 to just repeatedly find a maximum matching and assign the items accordingly until all items are allocated. 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 similar valuations, wherefore it may be beneficial for an agent to leave items to other agents if those agents cannot expect many more valuable items in the future.

What is stimmt das?

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 the valuations by agent i 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, too, does 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

Algorithm 1: SMatch for the Asymmetric Additive NSW problem

Input: set $\mathcal{A} = \{1, \dots, n\}$ of agents with weights $\eta_i \forall i \in \mathcal{A}$, set $\mathcal{G} = \{1, \dots, m\}$ indivisible items, additive valuations $v_i : \mathcal{P}(\mathcal{G}) \to \mathbb{R}_0^+$ where $v_i(\mathcal{S})$ is the valuation of agent $i \in \mathcal{A}$ for each item set $\mathcal{S} \subset \mathcal{G}$

Output: $\frac{1}{2n}$-approximation $\boldsymbol{x} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)$ of an optimal allocation

$\boldsymbol{x}_i \leftarrow \emptyset \quad \forall i \in \mathcal{A}$
$u_i \leftarrow v_i(\mathcal{G}_i^{2n+1}, \dots, \mathcal{G}_i^m) \quad \forall i \in \mathcal{A}$
$\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(j) + u_i/n) \mid i \in \mathcal{A}, j \in \mathcal{G} \,\}$ ▷ *edge weights*
$G \leftarrow (\mathcal{A}, \mathcal{G}, \mathcal{W})$ ▷ *bipartite graph*
$\mathcal{M} \leftarrow$ max_weight_matching(G)
$\boldsymbol{x}_i \leftarrow \{\, j \mid (i,j) \in \mathcal{M} \,\} \quad \forall i \in \mathcal{A}$ ▷ *allocate according to matching*
$\mathcal{G}^{\text{rem}} \leftarrow \mathcal{G} \setminus \{\, j \mid (i,j) \in \mathcal{M} \,\}$ ▷ *remove allocated goods*
while $\mathcal{G}^{\text{rem}} \neq \emptyset$ **do**
  $\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(j) + v_i(\boldsymbol{x}_i)) \mid i \in \mathcal{A}, j \in \mathcal{G}^{\text{rem}} \,\}$
  $G \leftarrow (\mathcal{A}, \mathcal{G}^{\text{rem}}, \mathcal{W})$
  $\mathcal{M} \leftarrow$ max_weight_matching(G)
  $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i \cup \{\, j \mid (i,j) \in \mathcal{M} \,\} \quad \forall i \in \mathcal{A}$
  $\mathcal{G}^{\text{rem}} \leftarrow \mathcal{G}^{\text{rem}} \setminus \{\, j \mid (i,j) \in \mathcal{M} \,\}$
end while
return $\boldsymbol{x}$

and is defined as

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

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 set of items assigned to agent i in previous matchings. The addend $v_i(\boldsymbol{x}_i)$ could lead to better allocations in applications, but does not improve the approximation factor asymptotically.

For convenience, we order the items in the final allocation $\boldsymbol{x}_i = \{h_i^1, \dots, h_i^{\tau_i}\}$ of agent i by the order in which they were assigned, so that item h_i^t is assigned after the t-th matching. To calculate the approximation factor of SMatch, we first need to establish a lower bound on the valuation of single items.

Lemma 1. *For each agent i and their final allocation $\boldsymbol{x}_i = \{h_i^1, \dots, h_i^{\tau_i}\}$, the item h_i^t is worth at least as much as the overall tn-th most highly valued item $\mathcal{G}_i^{tn}$, i. e. $v_i(h_i^t) \ge v_i(\mathcal{G}_i^{tn})$, for all $t = 1, \dots, \tau_i$.*

Proof. At the start of the t-th iteration, 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 iterations since at most n items are assigned in each iteration. During the t-th iteration, at most $n-1$ more of those highly valued items could be assigned to all other agents $i' \neq i$, leaving at least one item in $\mathcal{G}_i^1, \dots, \mathcal{G}_i^{tn}$ unassigned. Since $v_i(\mathcal{G}_i^k) \ge v_i(\mathcal{G}_i^{tn})$ for all $k \le tn$ 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. $v_i(h_i^2, \dots, h_i^{\tau_i}) \ge u_i/n$.

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 $k \in \{0, \dots, n-1\}$ and, consequently, its valuation $v_i(h_i^t)$ is at least as high as the mean valuation $\frac{1}{n} v_i(\mathcal{G}_i^{tn}, \dots, \mathcal{G}_i^{tn+n-1})$. Further, it holds $\tau_i n + n \geq m$ since each agent receives items for at least $\lfloor \frac{m}{n} \rfloor \geq \frac{m}{n} - 1$ rounds. Together, this yields

$$v_i(h_i^2, \dots, h_i^{\tau_i}) = \sum_{t=2}^{\tau_i} v_i(h_i^t) \geq \sum_{t=2}^{\tau_i} \frac{1}{n} v_i(\mathcal{G}_i^{tn}, \dots, \mathcal{G}_i^{tn+n-1}) \tag{2}$$

$$\geq \frac{1}{n} v_i(\mathcal{G}_i^{2n}, \dots, \mathcal{G}_i^{m-1}) \geq \frac{1}{n} v_i(\mathcal{G}_i^{2n+1}, \dots, \mathcal{G}_i^{m}) = \frac{u_i}{n} \tag{3}$$

with the last inequality stemming from $v_i(\mathcal{G}_i^{2n}) \geq v_i(\mathcal{G}_i^m)$. □

This allows us to calculate a lower bound on the approximation factor of SMatch.

Theorem 1. *SMatch has an approximation factor of* $2n$.

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

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

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

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

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 allocation $\boldsymbol{x}_i^* = \{g_i^1, \dots, g_i^{\tau_i^*}\}$ yields the even lower bound

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

Recall the definition of u_i from eq. (1). Consider a slightly modified variant:

$$u_i = \min_{\substack{\mathcal{S} \subset \mathcal{G} \\ |\mathcal{S}| \leq 2n}} \{v_i(\mathcal{G} \setminus \mathcal{S})\} \quad \text{or, alternatively,} \quad u_i = v_i(\mathcal{G} \setminus \mathcal{S}) \text{ with } \mathcal{S} := \arg\min_{\substack{\mathcal{S} \subset \mathcal{G} \\ |\mathcal{S}| \leq 2n}} \{v_i(\mathcal{G} \setminus \mathcal{S})\} \tag{8}$$

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

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

We get the lower bound $v_i(g_i^1) \geq \frac{1}{2n} v_i(\mathcal{S}^*)$ from a similar argument as in the proof of lemma 2. Further, it holds $u_i = v_i(\mathcal{G} \setminus \mathcal{S}) \geq v_i(\boldsymbol{x}_i^* \setminus \mathcal{S}) \geq v_i(\boldsymbol{x}_i^* \setminus \mathcal{S}^*)$. We can substitute these into eq. (7) and prove the theorem thereby:

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

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

□

3 RepReMatch

-

Algorithm 2: RepReMatch for the Asymmetric Submodular NSW problem

Input: set $\mathcal{A} = \{1, \dots, n\}$ of agents with weights $\eta_i \forall i \in \mathcal{A}$, set $\mathcal{G} = \{1, \dots, m\}$ indivisible items, additive valuations $v_i \colon \mathcal{P}(\mathcal{G}) \to \mathbb{R}_0^+$ where $v_i(\mathcal{S})$ is the valuation of agent $i \in \mathcal{A}$ for each item set $\mathcal{S} \subset \mathcal{G}$

Output: $\frac{1}{2n \log n}$-approximation $\boldsymbol{x}^{\mathrm{III}} = (\boldsymbol{x}_1^{\mathrm{III}}, \dots, \boldsymbol{x}_n^{\mathrm{III}})$ of an optimal allocation

Phase I:

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

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

for $t = 0, \dots, \lceil \log n \rceil - 1$ **do**

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

    $\mathcal{W} \leftarrow \{\, \eta_i \cdot \log(v_i(j)) \mid i \in \mathcal{A}, j \in \mathcal{G} \,\}$

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

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

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

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

  end if

end for

Phase II:

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

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

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

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

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

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

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

end while

Phase III:

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

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

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

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

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

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

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

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

decide on $\boldsymbol{x}$ or $\boldsymbol{x}^{\mathrm{III}}$; discrepancies in orig paper in def of $\boldsymbol{x}^{\mathrm{III}}$!