AN APPROXIMATION ALGORITHM FOR MAX-MIN FAIR ALLOCATION OF INDIVISIBLE GOODS*

ARASH ASADPOUR AND AMIN SABERI †

Abstract. In this paper, we give the first approximation algorithm for the problem of max-min fair allocation of indivisible goods. An instance of this problem consists of a set of k people and m indivisible goods. Each person has a known linear utility function over the set of goods which might be different from the utility functions of other people. The goal is to distribute the goods among the people and maximize the minimum utility received by them.

The approximation ratio of our algorithm is $\Omega(\frac{1}{\sqrt{k}\log^3 k})$. As a crucial part of our algorithm, we design and analyze an iterative method for rounding a fractional matching on a tree which might be of independent interest. We also provide better bounds when we are allowed to exclude a small fraction of the people from the problem.

1. Introduction. Fair division, also known as the cake cutting problem, has been studied extensively, mostly with a measure theoretic or combinatorial perspective in mathematics literature since the 1950s [26]. More recently this problem has been brought to the attention of the computer science community, in which the emphasis is more on designing efficient algorithms [24, 21, 22]. Moving beyond the metaphor of cake, the focus of most of these works is on practical cases where the goods are less divisible. An interested reader can consult [9] for a long list of applications of these problems.

In this paper, we study the problem of max-min fair allocation of indivisible goods. In our setting, we are allocating m goods to k persons. Each person i has a nonnegative integer valuation u_{ij} for good j. We assume that the valuation functions are linear; i.e. $u_{i,C} = \sum_{j\in C} u_{ij}$ for any set C of goods. The goal is to allocate each good to a person such that the least happy person is as happy as possible. More formally, we want to maximize $\min_i u_{i,C_i}$, where C_i is the set of goods allocated to person i.

This problem is very similar in nature to job scheduling problems especially minimum makespan scheduling [23]. However, applications of techniques such as rounding the assignment linear program [8, 16] has not yielded algorithms with an approximation ratio better than $O(1/k)$ for this problem so far. Namely, the assignment LP used in [23] yields an additive approximation of $\max_{ij} u_{ij}$ [8]; i.e. it can be used to find a solution of value at least $\mathrm{OPT} - \max_{ij} u_{ij}$, where OPT is the value of the optimal solution. Unfortunately, it offers no multiplicative approximation guarantee in the most challenging cases of the problem when $\mathrm{OPT} \leq \max_{ij} u_{ij}$. In the case of subadditive utility functions, Khot and Ponnuswami [20] provide a $(2k-1)$-approximation algorithm. Bezakova and Dani [8] showed that this problem is hard to approximate within a factor better than $\frac{1}{2}$. (In fact, this is also the best hardness result known for the general problem.) The most notable progress was made by Bansal and Sviridenko [6], who gave an $\Omega(\log\log\log k/\log\log k)$-approximation schema for the special case of "restricted assignment" in which for all i, j we have $u_{ij} \in \{0, u_j\}$. Their method involves rounding a certain type of linear programming relaxation of the problem (known as a *configuration* LP, which we will explain shortly). Later,

*A preliminary version of this paper appeared in the *Proceedings of the* 39*th Annual ACM Symposium on Theory of Computing (STOC)*, 2007, pp. 114-121.

†Department of Management Science and Engineering, Stanford University, Stanford, CA 94305. Email:{asadpour,saberi}@stanford.edu.

Feige [13] showed that the integrality gap of the LP used in [6] is in fact bounded by some large enough constant. Recently, Asadpour, Feige, and Saberi [3] showed that a local search algorithm can round the solution of the configuration LP to a $\frac{1}{5}$-approximate solution. However, the local search is not guaranteed to find a local optimum in polynomial time. Therefore, finding a constant factor approximation of the problem in the restricted case lies in the complexity class of Polynomial-time Local Search (PLS) [19].

In this work we introduce the first nontrivial approximation algorithm for the max-min fair allocation problem in the general case. Our algorithm finds an allocation in which the utility of every person is at least $\Omega(\frac{1}{\sqrt{k}\log^3 k})$ of the optimum. In [6], it is shown that the integrality gap of the configuration LP for this problem is $\sqrt{k}$. Therefore, up to some poly-log factor, this is the best that we can expect to get via this LP relaxation.

In the configuration program formulation of the problem, there is an indicator variable $x_{i,C}$ for assigning a valid bundle C to a person i. A valid bundle for person i is a bundle that makes person i (approximately) happy. In our problem, a valid bundle is either a *matching* bundle consisting of a single good with high value for i or a *flow bundle* that contains several goods with relatively low value for person i. Here, we will explain briefly how we round the solution of this LP. The concrete description of our algorithm can be found in Section 2.

Consider the bipartite graph defined by the solution of the LP in which the weights of the edges indicate what fraction of a good is allocated to each person. Also consider the two polytopes defined by the fractional assignment of goods to people via matching and flow edges. Both of these polytopes are very well behaved. However, they are defined on the same vertex set and they can not be rounded separately. In fact, the most critical part of the algorithm is to partition the vertices of the graph between the two polytopes. Once we correctly decide which persons should be satisfied with matching edges, it is possible to allocate the rest of the goods to the unmatched persons with a small loss.

We will use a randomized method for rounding the matching edges. For every vertex v corresponding to a good or a person, let m_v be the total fraction of matching edges incident to v. We round the fractional matching in such a way that the probability that a vertex v is saturated by the resulting integral matching is m_v. There are many ways to do this. For example, one can write the fractional matching as a convex combination of a few integral matchings and pick one of the integral matchings at random with probability proportional to its weight, e.g. using Birkhoff-Von Neumann decomposition. However, such a rounding scheme will impose lots of undesirable dependencies between different vertices and is not appropriate for our goal.

Instead, we show that without loss of generality we can assume that matching edges form a forest. Then we round the edges of the resulting forest iteratively. The description of this part of the algorithm is given in Section 3. Our rounding method tries to avoid imposing additional dependencies between vertices. In fact, we can prove that among all the feasible distributions on the matchings of the forest, the *entropy* of the distribution defined by our rounding algorithm is the highest (for the formal discussion, see Section 5.2).

The main part of the analysis of our algorithm is to show that after allocating a part of the goods defined by the matching, there will be enough goods left for every unsaturated person. This is done mainly by showing concentration results on the number of elements released in every bundle. We use generalizations of Azuma-

Hoeffding inequality on a martingale defined by the process of our algorithm. This part of the analysis is tricky because the expected values of the random variables of interest are sometimes smaller than the maximum step size in the evolution of the martingale. In Section 4.1 we prove our concentration result.

Finally, in Section 5.1 we extend our algorithm to find a solution in which more than a $1 - O(\frac{k}{\alpha^2 \log k})$ fraction of people receive a bundle with value at least $\frac{\text{OPT}}{\alpha \log^3 k}$, where OPT is the value of the optimal solution in the case that we want to satisfy all the people and α is any arbitrary chosen parameter greater than $48 \log k$.

Recently, as a follow-up to our work, Bateni, Charikar, and Guruswami [7] designed an approximation algorithm for a special case of this problem, where the integrality gap of the configuration LP is still known to be large. This special case consists of instances in which all the values are within the set $\{0, 1, \infty\}$. Moreover, each item can have value ∞ for at most one person. They show that the integrality gap of the configuration LP in these instances can be as big as $\Omega(k^{1/6})$. Their algorithm is based upon working with a simpler (and polynomial size) linear programming relaxation of the problem and applying multiple rounds of Sherali-Adams lift-and-project on it. They achieve an $O(m^{-\epsilon})$-approximation in $m^{O(1/\epsilon)}$ time for this special case. They also provide a 4-approximation for another special case of the max-min fair allocation problem in which every item has a nonzero value for at most two people. They also show that none of these problems can be approximated with a factor better than 2 unless P = NP.

Later, Chakrabarty, Chuzhoy, and Khanna [10] improved on the results of [7] and obtained an $O(m^{-\epsilon})$-approximation in $m^{O(1/\epsilon)}$ time and, consequently, a poly-logarithmic approximation in quasi-polynomial time for the general problem. Note that their algorithm outperforms ours in the case that $m = \text{poly}(k)$. They also showed a $(2 + \epsilon)$-approximation for the other special case with at most two nonzero utilities for each item. The running time of this algorithm is $\text{poly}(m, 1/\epsilon)$.

2. Description of the Algorithm. Fix a real number T, and suppose that we want to see if it is possible to do the allocation such that every person i receives a bundle with value at least T. We model the problem of finding such an allocation by a configuration LP similar to that of [6]. A *configuration* is a subset of goods. A configuration C is called *valid* for person i if either $u_{i,C} \geq T$ or it contains a single good j with $u_{ij} \geq T/\sqrt{k} \log^3 k$. Let $\mathcal{C}(i, T)$ denote the set of all valid configurations corresponding to person i with respect to T. Now, we can model the problem by the following integer program:

$$\begin{aligned}
&\forall j, \quad \sum_{C \ni j} \sum_{i} x_{i,C} \leq 1, \\
&\forall i, \quad \sum_{C \in \mathcal{C}(i,T)} x_{i,C} = 1, \\
&\forall i, C, \quad x_{i,C} \in \{0, 1\}.
\end{aligned}$$

The indicator variable $x_{i,C}$ shows if the configuration C is allocated to person i or not. The first and second sets of constraints respectively ensure that we do not over-allocate any good and that every person has received at least one bundle with

high enough value. The configuration LP relaxation of the problem is as follows:

$$\begin{aligned} &\forall j, \quad \sum_{C \ni j} \sum_{i} x_{i,C} \leq 1, \qquad (1)\\ &\forall i, \quad \sum_{C \in \mathcal{C}(i,T)} x_{i,C} = 1,\\ &\forall i, C, \quad x_{i,C} \geq 0. \end{aligned}$$

Note that if the LP is feasible for some T_0, then the corresponding LP for all $T \leq T_0$ is also feasible. Let OPTLP be the maximum of all such values of T (it can be shown that such a maximum exists). On the other hand, every feasible integral allocation is a feasible solution of its corresponding configuration LP. Therefore, OPTLP $\geq$ OPT. Also, the value of OPTLP and a feasible solution corresponding to OPTLP can be approximated within any desired degree of accuracy in polynomial time, as shown in [6].

It is shown in [6] that the integrality gap of the above LP is $\Omega(\sqrt{k})$. Here we propose an approach for rounding in which each person will receive a bundle with value at least $\Omega(\frac{T}{\sqrt{k}\log^3 k})$.

We define a weighted bipartite graph G, with the vertex set $A \cup B$ corresponding to persons and goods, respectively. There is an edge between vertices corresponding to person $i \in A$ and good $j \in B$ if a configuration C containing j is fractionally assigned to i. Define $w_{i,j} = \sum_{C \ni j} x_{i,C}$; i.e. $w_{i,j}$ is the fraction of good j that is allocated to person i by the fractional solution of LP (1). A good j is *big* for a person i if $u_{ij} \geq \frac{T}{1440\sqrt{k}\log^3 k}$; otherwise, it is *small*. The edge $\{i, j\}$ in the former case is called a *matching* edge, whereas it is called a *flow* edge in the latter. Let $\mathcal{M}$ and $\mathcal{F}$ represent the set of matching and flow edges, respectively. For each vertex $v \in A \cup B$, let m_v be the total fraction of the matching edges incident to it. Also, define $f_v = 1 - m_v$.

Our algorithm applies a two-stage randomized rounding to the solution of the LP at the top of this page. In the first stage we pick a random matching from $\mathcal{M}$ in a way that with probability m_v the vertex v is in the matching. We will explain the details of the matching algorithm in the next section.

In the second stage every remaining person i selects one of her bundles C consisting of small goods with probability proportional to $x_{i,C}$. From that bundle, she will claim the set of goods not saturated by the matching. Note that each person i would not be saturated by the matching with probability $1 - m_i = f_i$. Hence, each such person i will select bundle C with probability $x_{i,C}/f_i$.

Later, we will show that with high probability there will be enough goods left in every bundle; i.e. the valuation of i for the goods left in the bundle will remain above $T/\sqrt{k}\log^2 k$.

Note that every good might be claimed by several persons. In the last step, we give every good to one of the persons who have claimed that good uniformly at random. Our algorithm is represented in the Figure 1.

3. Finding a Random Matching. It is possible to construct several examples to show that rounding of matching edges should be done with a lot of care. For example, a close inspection of Bansal and Sviridenko's [6] example for the integrality gap shows that there is very little flexibility in terms of the set of vertices that should be saturated in the matching when we round the fractional solution. See Figure 4 in the appendix.

Algorithm 1: The Main Algorithm.

1. Do a binary search to guess T, the value of the optimal solution. Solve LP (1).
   Find the set $\mathcal{M}$ and compute m_v and f_v for every vertex in G.
2. Select a random matching from edges in $\mathcal{M}$ using Algorithm 2 such that for every $v \in A \cup B$ the probability that v is saturated by the matching is $m_v = 1 - f_v$.
3. Let every person i who is not matched yet select a bundle C consisting of small goods with probability $x_{i,C}/f_i$ and claim the goods in that bundle (except the ones that are assigned in the previous step).
4. Some of the goods will be claimed by more than one person. Resolve these conflicts as follows: Define a fractional solution that divides each good equally between all the people who have claimed it. Use the additive algorithm of [8] to round this fractional solution. (The details are described in Section 4.3.)

FIG. 1. *: The Main Algorithm*

We want that each vertex v in the graph is saturated by the matching with probability m_v. As we pointed out before, we also want that our algorithm does not impose unnecessary dependencies between vertices. In [2] Arora, Frieze, and Kaplan propose a method for "mixing" the integral matchings to decrease this dependency but it seems that their method is not accurate enough for our purpose. We will propose a different scheme here. Our scheme takes advantage of the fact that without loss of generality, we can assume that $\mathcal{M}$ is a forest. The following lemma, gives a proof for this simplifying observation. This observation has been made before in similar settings, most notably in [23].

LEMMA 1. *It is possible to adjust the weights of the edges in $\mathcal{M}$ such that for every vertex v, $\sum_u w_{u,v}$ remains the same, and the set of edges with non-zero weight form a forest.*

Proof. Suppose that there is a cycle in $\mathcal{M}$ with edges $e_1, e_2, \cdots, e_{2a}$. Let e_1 be an edge with minimum weight among all those edges, namely w, then construct a new solution that is the same as this solution on all other edges, but in these $2a$ edges the weight of odd indexed edges is decreased by w and the weight of the others is increased by w. This process does not change the sum of all edges adjacent to each vertex eventually reduces $\mathcal{M}$ to a forest. □

Our algorithm for selecting a random matching in the obtained forest is shown in Figure 2. It iteratively picks a vertex u that has not been processed yet and matches it with one of its neighbors v with probability w^i_{uv} or none of them with probability $1 - p^i_u$, where w^i_{uv} is the weight of edge $\{i, j\}$ before taking the step $i + 1$, and $p^i_u = \sum_v w^i_{uv}$. Then it updates the weights in the whole forest.

Here is the informal intuition about our update rule: If for each u, v, w_{uv} is interpreted as the probability of picking the edges $\{u, v\}$ in the matching, then we should inform all the edges in the forest about the decision we made at step i. The update rule does this via updating the probability by which each edge is selected in the matching.

Let $\mathbf{F}_i$ be the σ-field corresponding to the probability distribution that our algorithm imposes over matchings, before step $i + 1$ is taken. The crucial observation

Algorithm 2 [Picking a Random Matching]

1. Eliminate cycles of $\mathcal{M}$ using the technique mentioned in the proof of Lemma 1.
2. Pick a vertex v that has not been processed yet.
3. For all edge $e \in \mathcal{M}$: $w_e^{\text{NEW}} \leftarrow w_e$.
   (a) Either pick one of edges adjacent to v, namely e, with probability equal to the weight of these edges and:
      i. $w_e^{\text{NEW}} \leftarrow 1$
      ii. For all edges e' adjacent to e: $w_{e'}^{\text{NEW}} \leftarrow 0$.
   or with the remaining probability for all edges e adjacent to v: $w_e^{\text{NEW}} \leftarrow 0$
   (b) For all other edges e_0 in the same connected component as v, find the unique path that connects e_0 to the first edge whose weight has become zero in previous step, namely e_l. Let this path be $e_0, e_1, ..., e_l$.

$$w_{e_0}^{\text{NEW}} \leftarrow \left[1 + (-1)^{l-1} \frac{w_{e_1} \cdots w_{e_l}}{(1 - w_{e_1}) \cdots (1 - w_{e_l})}\right] w_{e_0}. \quad (2)$$

   (c) For all edges $w_e \leftarrow w_e^{\text{NEW}}$. Remove the edges with weight 0.

FIG. 2. *Picking a Random Matching*

about this algorithm is stated in the following.

LEMMA 2. *For all edges e' and for all i, $E[w_{e'}^i | \boldsymbol{F}_{i-1}] = w_{e'}^{i-1}$.*

Proof. If the edge e' is adjacent to the vertex v that we select in the step i then with probability $w_{e'}^{i-1}$ we change its weight to one and with probability $w_{e'}^{i-1}$ to zero and the claim holds. Also if e' is adjacent to an edge e which itself is adjacent to v, we have

$$E[w_{e'}^i | \mathbf{F}_{i-1}] = 0 \times w_e^{i-1} + \left[1 + \frac{w_e^{i-1}}{1 - w_e^{i-1}}\right] w_{e'}^{i-1} \times (1 - w_e^{i-1}) = w_{e'}^{i-1}$$

For all other cases, if there is not a path with positive weight intermediate edges between e' and the vertex v before this step, then $w_e^i = w_e^{i-1}$ and we are done. Otherwise let $e_0 = e', e_1, \cdots, e_l$ be such a path between e' and vertex v. Thus,

$$\begin{aligned} E[w_{e'}^i | \mathbf{F}_{i-1}] = & \left[1 + (-1)^{l-2} \frac{w_{e_1}^{i-1} w_{e_2}^{i-1} \cdots w_{e_{l-1}}^{i-1}}{(1 - w_{e_1}^{i-1})(1 - w_{e_2}^{i-1}) \cdots (1 - w_{e_{l-1}}^{i-1})}\right] w_{e'}^{i-1} \times w_{e_l}^{i-1} \\ & + \left[1 + (-1)^{l-1} \frac{w_{e_1}^{i-1} w_{e_2}^{i-1} \cdots w_{e_l}^{i-1}}{(1 - w_{e_1}^{i-1})(1 - w_{e_2}^{i-1}) \cdots (1 - w_{e_l}^{i-1})}\right] w_{e'}^{i-1} \times (1 - w_{e_l}^{i-1}) \\ & = w_{e'}^{i-1} \end{aligned}$$

□

The following corollary is an immediate result of Lemma 2.

COROLLARY 3. *The probability that a vertex $v \in A \cup B$ is saturated by the matching generated by Algorithm 2 is m_v.*

Another important characteristic of our method which will be used in our analysis, is stated in the following lemma.

LEMMA 4. *Our algorithm is* Order Independent*; It means that independent from the ordering, it always gives us a matching M with probability $p(M)$ that only depends on the initial weights of edges in the forest and the matching M itself.*

Proof. Straightforward calculation shows that in an ordering, swapping two consecutive vertices will not change the distribution of the weights of the edges after the two steps corresponding to these vertices. The lemma then follows by induction. We omit the details here, but for a formal and entirely different proof one can see Remark 2 in Section 5.2. □

4. Analysis. In this part we will prove that the event that our algorithm does not find an integral solution with value $\frac{T}{\sqrt{k}\log^3 k}$ is very rare. We will do that in three steps.

First, we show that for a subset $S \subseteq A$ or $S \subseteq B$, the distribution of the number of vertices saturated from that subset by the random matching is concentrated around its expected value $\sum_{v\in S} m_v$. This is proved in Section 4.1.

Then we will proceed to show that after allocating the goods defined by the matching, there will be enough goods left for every unsaturated person. In particular, the probability that in step 3 of the algorithm a person claims a bundle in which the value of the remaining goods is less than $\frac{T}{40\sqrt{k}\log^2 k}$ is very small. This is done in Section 4.2

Finally, we will prove that with high probability a large portion of the goods will be claimed by no more than $O(\log k)$ persons. Due to this fact, we show in Section 4.3 that one can allocate the goods to the unsaturated people such that every person receives about $1/\log k$ of the her claimed bundle.

4.1. Concentration Results for Algorithm 2. Fix a subset $S \subseteq A$. We will prove that the number of vertices in S saturated by the random matching is concentrated around its expected value $\sum_{v\in S} m_v$. The proof for a subset $S \subseteq B$ will be the same. Define the random variable $X_i = \sum_{u\in S} p_u^i$. The next corollary is an immediate result of Corollary 3.

COROLLARY 5. *The random variables $X_0, X_1, \cdots, X_{END} = X$ define a martingale.*

Note that X denotes the number of vertices that will be saturated at the end of Algorithm 2 from subset S and $EX = X_0$. Observe that the length of jumps can be as big as 1, and our martingale may have k steps. On the other hand, EX_0 might be so small that the standard Azuma-Hoeffding's inequality could not be applied to get the desired bound. Instead, we will use an extension of the Azuma-Hoeffding inequality from [11].

A *filter* $\mathbf{G}$ is an increasing chain of σ-subfields

$$\{0, \Omega\} = \mathbf{G}_0 \subset \mathbf{G}_1 \subset \cdots \subset \mathbf{G}_n = \mathbf{G}.$$

The following result can be found in [11].

THEOREM 6. ***[Chung and Lu]*** *Let martingale Y be associated with the filter $\boldsymbol{G}$ and a sequence of random variables $Y_0, Y_1, \cdots, Y_n$ satisfying $Y_i = E(Y|\boldsymbol{G}_i)$ and, in particular $Y_0 = EY$ and $Y_n = Y$. Assume Y satisfies*

1. *$Var(Y_i|\boldsymbol{G}_{i-1}) \le \sigma_i^2$, where $1 \le i \le n$;*
2. *$|Y_i - Y_{i-1}| \le M$, for $1 \le i \le n$.*

Then, we have

$$\Pr(Y - EY \geq \lambda) \leq e^{-\lambda^2/2(\sum_{i=1}^{n} \sigma_i^2 + M\lambda/3)}. \tag{3}$$

In order to employ the above theorem to get the desired concentration result, we would need to bound both the length of jumps and also variances of random variables in the martingale. Lemma 7 will bound the maximum change in random variables in every single step. Lemma 11 will provide such a bound on the variance.

LEMMA 7. *For all i, we have: $|X_i - X_{i-1}| \leq 1$.*

We need to make some preliminary observation about the process of Algorithm 2 in order to prove the above lemma. Because each X_i is the sum of some p_u^i's, we first bound the jumps in the values of p at each step. The following Lemma bounds the change in p_u^i with respect to p_u^{i-1}.

LEMMA 8. *Suppose that vertex $v \in A$ is processed during the step i of Algorithm 2. Let $u \in A$ be an arbitrary vertex, which is also in the same connected component of the current forest as v. Let $v' \in B$ be the last vertex before v on the unique path from u to v in this forrest. If v is matched by the edge $\{v', v\}$ in step i (Case I), then we have*

$$p_u^i = p_u^{i-1} - L_{uv'}(1 - p_u^{i-1}),$$

Otherwise (Case II), we have

$$p_u^i = p_u^{i-1} + L_{uv}(1 - p_u^{i-1}).$$

Where for every pair (u, v) we have $L_{uv} = \prod_e \frac{w_e^{i-1}}{1-w_e^{i-1}}$, where the product is taken over all the edges in the unique path between u and v (see Figure 3).

REMARK 1. *Note that both u and v are in the same side of the bipartite graph (i.e. $u, v \in A$). Therefore, the distance between u and v is ensured to be even.*

Proof. Let u' be the first vertex after u on the path from u to v and let $e' = (u, u')$. Also Let N be the set of other edges adjacent to u. We will prove the first equality. The second equality can be proved the same way.

Suppose that the edge $\{v', v\}$ is picked in the step i. From relation (2) in Algorithm 2 we have the following:

$$w_{e'}^i = (1 - L_{u'v'})w_{e'}^{i-1}, \forall e \in N : w_e^i = (1 + L_{uv'})w_e^{i-1}.$$

Hence,

$$\begin{aligned}
p_u^i &= (1 - L_{u'v'})w_{e'}^{i-1} + \sum_{e \in N}(1 + L_{uv'})w_e^{i-1} \\
&= (1 - L_{u'v'})w_{e'}^{i-1} + (1 + \frac{w_{e'}^{i-1}}{1 - w_{e'}^{i-1}} L_{u'v'}) \sum_{e \in N} w_e^{i-1} \\
&= w_{e'}^{i-1} + \sum_{e \in N} w_e^{i-1} - L_{u'v'} \left(w_{e'}^{i-1} - \frac{w_{e'}^{i-1} \sum_{e \in N} w_e^{i-1}}{1 - w_{e'}^{i-1}} \right) \\
&= p_u^{i-1} - L_{u'v'} \frac{w_{e'}^{i-1}(1 - w_{e'}^{i-1} - \sum_{e \in N} w_e^{i-1})}{1 - w_{e'}^{i-1}} = p_u^{i-1} - L_{uv'}(1 - p_u^{i-1}).
\end{aligned}$$

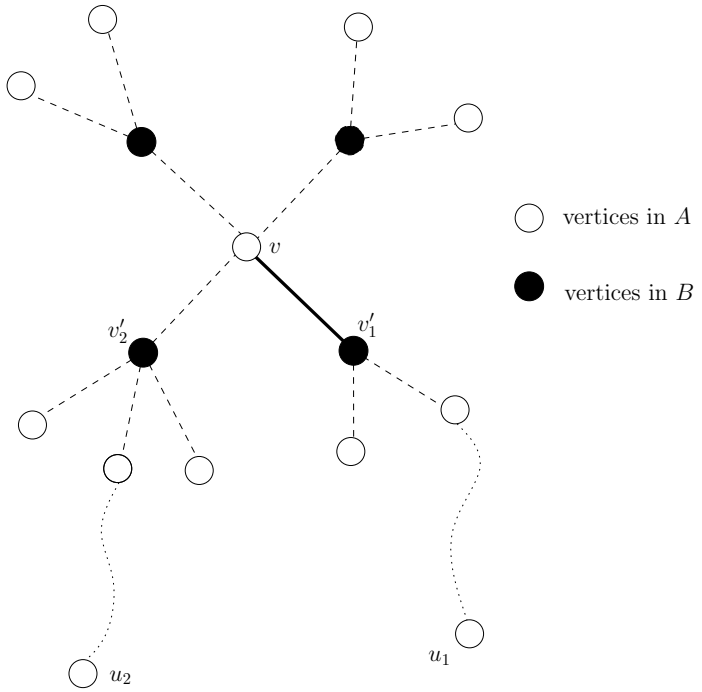

FIG. 3. *Vertex $v \in A$ is processed during the step i of Algorithm 2. Vertices u_1 and u_2 are two arbitrary vertices in A. Moreover, $v'_1 \in B$ (resp. $v'_2 \in B$) is the last vertex before v on the path from u'_1 (resp. u'_2) to v. At step i we decide to match v to v'_1. Hence, Lemma 8 classifies u_1 as in Case I and u_2 as in Case II.*

The second and forth equality follows from the fact that by definition

$$L_{uv'} = L_{u'v'} \frac{w_{uu'}^{i-1}}{1 - w_{uu'}^{i-1}}.$$

The fifth equality is immediate by the definition of p_u^{i-1}. The other two equalities are just algebraic manipulation. □

Now, we root the tree at v. Let $\hat{j}$ denote the father of j and C_j denote the set of grandchildren of j. For each vertex j, define T_j to be the set of all vertices in A which lie in the subtree rooted at j.

LEMMA 9. *For all the vertices $r \neq v \in A$,*

$$\sum_{u \in T_r} L_{uv}(1 - p_u^i) \leq L_{rv}(1 - w_{r\hat{r}}^i).$$

Proof. By induction on the height of vertices. For the leaves the left hand side of the inequality is zero and the right hand side is non-negative. Consider some vertex r with even edge distance from v. By the induction hypothesis, for all the vertices $j \in C_r$, we have

$$\sum_{u \in T_j} L_{uv}(1 - p_u^i) \leq L_{jv}(1 - w_{j\hat{j}}^i). \tag{4}$$

Now we compute the sum for the subtree rooted at vertex r. Note that by definition of T_r we have:

$$\sum_{u\in T_r} L_{uv}(1-p_u^i) = L_{rv}(1-p_r^i) + \sum_{j\in C_r}\sum_{u\in T_j} L_{uv}(1-p_u^i). \tag{5}$$

Plugging (4) into (5), we derive that

$$\sum_{u\in T_r} L_{uv}(1-p_u^i) \le L_{rv}(1-p_r^i) + \sum_{j\in C_r} L_{jv}(1-w_{j\hat{j}}^i).$$

We emphasize again that C_r is the set of the grandchildren of r, and for each $j \in C_r$, the father of j (and subsequently, a child of r) is denoted by $\hat{j}$. Hence, we conclude

$$\begin{aligned}
\sum_{u\in T_r} L_{uv}(1-p_u^i) &\le L_{rv}(1-p_r^i) + \sum_{j\in C_r} L_{jv}(1-w_{j\hat{j}}^i) \\
&= L_{rv}(1-p_r^i) + L_{rv}\sum_{j\in C_r} L_{rj}(1-w_{j\hat{j}}^i) && (6)\\
&= L_{rv}(1-p_r^i) + L_{rv}\sum_{j\in C_r} \frac{w_{\hat{j}r}^i w_{j\hat{j}}^i}{1-w_{\hat{j}r}^i} && (7)\\
&\le L_{rv}(1-p_r^i) + L_{rv}(p_r^i - w_{r\hat{r}}^i) && (8)\\
&= L_{rv}(1-w_{r\hat{r}}^i),
\end{aligned}$$

which completes the proof of the lemma. Note that (6) is a result of $L_{jv} = L_{jr}L_{rv}$, which holds by the definition of $L_{.,.}$ whenever the path from j to v in the tree contains r. Also, (7) is immediate by the definition of $L_{.,.}$. To derive (8), let S_r be the set of the children of r. For every $l \in S_r$ we have

$$p_u^i = \sum_{u\sim l} w_{ul}^i = w_{lr}^i + \sum_{j\in C_r:\hat{j}=l} w_{lj}^i \Rightarrow \sum_{j\in C_r:\hat{j}=l} w_{lj}^i \le 1 - w_{lr}^i. \tag{9}$$

Now, one can see that by definition of S_r we have

$$\sum_{j\in C_r} \frac{w_{\hat{j}r}^i w_{j\hat{j}}^i}{1-w_{\hat{j}r}^i} = \sum_{l\in S_r}\sum_{j\in C_r:\hat{j}=l} \frac{w_{\hat{j}r}^i w_{j\hat{j}}^i}{1-w_{\hat{j}r}^i} = \sum_{l\in S_r} \frac{w_{lr}^i}{1-w_{lr}^i}\left(\sum_{j\in C_r:\hat{j}=l} w_{lj}^i\right). \tag{10}$$

Combining (9) and (10) results in

$$\sum_{j\in C_r} \frac{w_{\hat{j}r}^i w_{j\hat{j}}^i}{1-w_{\hat{j}r}^i} \le \sum_{l\in S_r} w_{lr}^i = p_r^i - w_{r\hat{r}}^i,$$

which implies (8). □

LEMMA 10. *Let S be an arbitrary subset of A. Then*

$$\sum_{u\in S-\{v\}} p_u^i \le \sum_{u\in S} p_u^{i-1}.$$

Proof. During step i, the values of vertices in a different connected component as v do not change. Also by Lemma 8, the value p_u for an arbitrary vertex u will not increase (decrease) if the edge adjacent to v in its path to u is (is not) processed in the step i. Therefore, it is enough to prove the Lemma 10 only in the case that:

- The set S consists of all the vertices with even edge distance in the same connected component that v is, e.g. $S = T_v$.
- The vertex v is not saturated after step i.

We use Lemma 9 for all the vertices $u \in C_v$.

$$\begin{aligned}\sum_{u \in T_v - \{v\}} L_{uv}(1 - p_u^{i-1}) &\leq \sum_{j \in C_v} L_{jv}(1 - w_{j\hat{j}}^{i-1}) \\ &\leq \sum_{j \in C_v} \frac{w_{\hat{j}v}^{i-1} w_{j\hat{j}}^{i-1}}{1 - w_{\hat{j}v}^{i-1}} \\ &= p_v^{i-1}.\end{aligned}$$

It is sufficient to look at the case where $S = T_v$ and the vertex v is not saturated.

$$\begin{aligned}\sum_{u \in S - \{v\}} p_u^i &= \sum_{u \in S - \{v\}} \left(p_u^{i-1} + L_{uv}(1 - p_u^{i-1})\right) \\ &= \sum_{u \in S} p_u^{i-1} - p_v^{i-1} + \sum_{u \in S - \{v\}} L_{uv}(1 - p_u^{i-1}).\end{aligned}$$

Plugging in inequality (11) we conclude that

$$\sum_{u \in S - \{v\}} p_u^i \leq \sum_{u \in S} p_u^{i-1}.$$

□

Now, we are ready to prove Lemma 7.

Proof. **[Lemma 7]** The inequality $X_i \leq X_{i-1} + 1$ is implied from Lemma 10 and the fact that $p_v^i \leq 1$.

Now, we prove $X_i \geq X_{i-1} - 1$. Note that if vertex v is not saturated then by Lemma 8, for all $u \neq v$, $p_u^i \geq p_u^{i-1}$ so $X_i \geq X_{i-1} - p_v^{i-1} \geq X_{i-1} - 1$. Furthermore, if v is matched to a vertex v', then for all vertices $u \notin T_{v'}$, $p_u^i \geq p_u^{i-1}$. So we need to bound $M = \sum_{u \in T_{v'}} p_u^i - p_u^{i-1}$ by -1 from below.

M can take two different values based on whether or not v is matched to v'. If v is not matched to v',

$$\begin{aligned}\sum_{u \in T_{v'}} (p_u^i - p_u^{i-1}) &= \sum_{u \in T_{v'}} L_{uv}(1 - p_u^{i-1}) \\ &\leq \sum_{j : \hat{j} = v'} L_{jv}(1 - w_{jv'}^{i-1}) \qquad (11) \\ &= \sum_{j : \hat{j} = v'} \frac{w_{v'v}^{i-1} w_{jv'}^{i-1}}{1 - w_{v'v}^{i-1}} \qquad (12) \\ &= w_{vv'}^{i-1}.\end{aligned}$$

Inequality (11) is a result of Lemma 9. Also, equation (12) is immediate by definition of L_{jv}.

On the other hand, we know that $EM = 0$. Therefore, if v is matched to v', the value of M will be bounded from below by $w_{vv'}^{i-1}(w_{vv'}^{i-1} - 1)/(w_{vv'}^{i-1}) \geq -1$.

□

The next lemma provides the desired bound for the variance of random variables.

LEMMA 11. *If we process vertex $v \in S$ in the step i, then: $Var[X_i|\boldsymbol{F}_{i-1}] \leq 2p_v^{i-1}$.*

Proof. By definition

$$\begin{aligned}\mathrm{Var}[X_i|\mathbf{F}_{i-1}] &= E[(X_i - EX_i)^2|\mathbf{F}_{i-1}] \\ &= E[(X_i - X_{i-1})^2|\mathbf{F}_{i-1}] \\ &\leq E[|X_i - X_{i-1}||\mathbf{F}_{i-1}].\end{aligned}$$

The last inequality is justified by Lemma 7. Now since $E[X_i - X_{i-1}|\mathbf{F}_{i-1}] = 0$, we can write the last term as $2E[X_i - X_{i-1}|\mathbf{F}_{i-1}, X_i > X_{i-1}]\Pr(X_i > X_{i-1})$. By Lemma 10, X_i can be bigger than X_{i-1} only if v is saturated in step i, which happens with probability p_v^{i-1}. Therefore $E[(X_i - X_{i-1})^2|\mathbf{F}_{i-1}] \leq 2p_v^{i-1}$. □

The following Lemma establishes our main concentration result. It shows that the number of saturated vertices in any subset $S \in A$ is concentrated around its expected value. The corollary after that, provides a similar result about the subsets in the B side.

LEMMA 12. *For a subset $S \subseteq A$ and a given λ and the martingale X defined as above on the vertices of S, we have $\Pr(X - \mu \geq \lambda) \geq e^{-\frac{\lambda^2}{2(2\mu \log k + \lambda/3)}}$, where $\mu = EX$.*

Proof. Lemma 7 establishes the bounded difference property. Using Lemma 11, we need to bound the sum of the variances in terms of μ. By the order independence proved in Lemma 4, the probability distribution of the solution of Algorithm 2 does not change when we change the order in which the vertices are processed. We will choose an ordering that is more convenient for our analysis. In our ordering:

- The vertices in S are processed before other vertices.
- In each step $i, 1 \leq i \leq |S|$, we pick a vertex v, with minimum p_v^i among all the vertices that have not been processed yet, and process it. We show this vertex by v_i.

By Lemma 10, $\sum_{u \in S - \{v\}} p_u^i \leq \sum_{u \in S} p_u^{i-1}$. This implies that the sum of p_u^i's for all the vertices in S that are not processed yet, will not exceed $X_0 = \mu$. Therefore $p_{v_i}^{i-1} \leq \mu/(|S| - i + 1), 1 \leq i \leq |S|$. So we have:

$$\sum_{i=1}^{|S|} \mathrm{Var}[X_i|\mathbf{F}_{i-1}] \leq \sum_{i=1}^{|S|} 2p_{v_i^{i-1}} \leq 2\sum_{i=1}^{|S|} \mu/(|S| - i + 1) \leq 2\mu \log k.$$

Plugging this inequality in Theroem 6 completes the proof. □

COROLLARY 13. *For a subset $S \subseteq B$ and a given λ, let Z be a random variable denoting the number of vertices in S that are not saturated by Algorithm 2. Then $\Pr(Z - \mu \geq \lambda) \geq e^{-\frac{\lambda^2}{2(2\mu \log(k+|S|) + \lambda/3)}}$, where $\mu = EZ$.*

Proof. For each vertex $v \in S$ add a new dummy person and connect it to v with weight $1 - m_v$. The result immediately follows by applying Lemma 12 on set S consisting of all dummy vertices. □

4.2. Failing Pairs. Now we can study the process of rounding flow edges. As we stated in the algorithm, each person i who has not received a big good in Algorithm 2 will select one of her bundles, say C, with probability $x_{i,C}/f_i$. The probability that a person i does not receive a good in Algorithm 2, is $1 - m_i \leq f_i$ and hence the probability that she selects bundle C is at most $x_{i,C}$.

We call the pair (i, C) a *failing pair* if person i is not saturated at the end of Algorithm 2 and her valuation in the set of goods left in C after the matching is selected is less than $T/40\sqrt{k}\log^2 k$.

First, we will prove that the valuation of a person i in what will be left in a bundle C after Algorithm 2 is not much less than its expected value $R_i(C) = \sum_{j\in C} f_j u_{ij}$. Lemma 14 shows that if $R_i(C) > T/\sqrt{k}\log k$, then the probability that (i, C) is a failing pair is very small. But, the probability of observing at least one pair (i, C) with small value of $R_i(C)$ by union bound is at most $\sum x_{i,C}$, where the sum is taken over all such pairs. Finally, in Lemma 16 we will prove that this sum is bounded by $o(1)$. It concludes that with high probability after step 3 of Algorithm 2 each person unsaturated in step 2, will have a value at least $T/40\sqrt{k}\log^2 k$ from the unsaturated goods of her claimed bundle.

LEMMA 14. *If $R_i(C) \geq T/\sqrt{k}$, then the probability that (i, C) is a failing pair is at most $e^{-\log^2 k/40}$.*

Proof. We have to consider two cases. If C consists of many goods for which i has small valuation, i.e. if $\sum_{j\in C, u_{ij}\leq T/k^3} u_{ij} f_j \geq T/2\sqrt{k}$ then (i, C) can not be a failing pair at all. This is because the number of goods saturated in a matching is at most k. Therefore, the total valuation of person i for the goods lost to the matching from this interval is at most T/k^2. Her valuation for the rest of the goods is at at least $T/2\sqrt{k} - T/k^2 \geq T/\sqrt{k}\log k$.

Now suppose $\sum_{j\in C, u_{ij}\leq T/k^3} u_{ij} f_j < T/2\sqrt{k}$. We can find an integer $0 \leq \delta < 2.5\log k - 3\log\log k$, such that

$$\sum_{T2^\delta/k^3 \leq u_{ij} < T2^{\delta+1}/k^3} u_{ij} f_j \geq T/5\sqrt{k}\log k.$$

Let C' be the set of goods j such that $T2^\delta/k^3 \leq u_{ij} < T2^{\delta+1}/k^3$. We will have

$$\sum_{j\in C'} f_j \geq \frac{T}{5\sqrt{k}\log k} \times \frac{k^3}{T2^{\delta+1}} = \frac{k^{2.5}}{10\times 2^\delta \log k}.$$

Using Lemma 12 we can see that the probability that less $\frac{k^{2.5}}{20\times 2^\delta \log^2 k}$ of goods in C' are not saturated in the matching is at most $e^{-\log^2 k/40}$. To see this note that the RHS of the above inequality is at least $\log^2 k/10$. Now, if more than $\frac{k^{2.5}}{20\times 2^\delta \log^2 k}$ goods in C' are left unsaturated, then person i will have a valuation of at least $\frac{k^{2.5}}{20\times 2^\delta \log^2 k} \times \frac{2^\delta T}{k^3}$ for them which implies that (i, C) is not failing. □

The observation made in the next lemma is crucial to prove Lemma 16. It implies that for a failing pair (i, C), the major part of most of the goods in C is allocated via matching edges in the fractional solution.

LEMMA 15. *If for a pair (i, C) is so that $R_i(C) < T/\sqrt{k}$, then there exists a set, namely $M_i(C) \subseteq C$, such that $|M_i(C)| \geq \sqrt{k}\log^2 k/12$ and for all $j \in M_i(C)$ we have $f_j \leq 24\log k/\sqrt{k}$.*

Proof. We use the same idea applied in the proof of Lemma 14. Using a similar argument as before we can conclude that

$$\sum_{j\in C, u_{ij}\leq T/k^3} u_{ij} \leq T/2.$$

Therefore,

$$\exists r, \frac{T}{k^3} \leq r \leq \frac{T}{2\sqrt{k}\log^3 k} : \sum_{j \in C, r \leq u_{ij} < 2r} u_{ij} \geq \frac{T}{6\log k}.$$

For this bundle C, define C' to be the set of all goods j in C for which $r \leq u_{ij} < 2r$. For each $j \in C', u_{ij} \leq T/\sqrt{k}\log^3 k$. Therefore $|C'| \geq \frac{T/6\log k}{2r} = \sqrt{k}\log^2 k/6$.

Note that $R_i(C)$ and consequently, $R_i(M_i(C))$ are at most $T/\sqrt{k}$. It means that $r\sum_{j\in C'} f_j \leq R_i(C') \leq T/\sqrt{k}$, which implies $\sum_{j\in C'} f_j \leq \frac{T}{r\sqrt{k}}$.

But, for half of the goods j in C', f_j is less than twice the average. Therefore,

$$f_j \leq \frac{2T}{r\sqrt{k}|C'|} \leq \frac{24\log k}{\sqrt{k}}. \tag{13}$$

Those goods form the desired set $M_i(C)$. □

The lemma below summarizes the results of this section by proving that observing a failing pair is a very rare event.

LEMMA 16. *The probability that a failing pair (i, C) is observed in the third step of Algorithm 2 is $o(1)$.*

Proof. Let $\mathcal{C}$ be the set of all pairs (i, C) such that $R_i(C) < T/\sqrt{k}$.

Lemma 14 shows that with high probability (at least $1 - ke^{-\log^2 k/40} = 1 - o(1)$) every person i who has claimed a bundle C such that $(i, C) \notin \mathcal{C}$ will observe a value of at least $T/40\sqrt{k}\log k$ in C after the matching is selected. Therefore, it suffices to prove that the probability that some person i claims a bundle C such that $(i, C) \in \mathcal{C}$ is $o(1)$.

By union bound, this probability is bounded by $\sum_{(i,C)\in\mathcal{C}} x_{i,C}$. Let N be the set of all goods that belong to $M_i(C)$ for a failing pair $(i, C) \in \mathcal{C}$.

$$\sum_{j\in N}(1 - f_j) = \sum_{j\in N} m_j \leq k \implies |N|(1 - \frac{24\log k}{\sqrt{k}}) \leq k \implies |N| < 2k$$

By equation (13) $\sum_{j\in N} f_j \leq |N| \times \frac{24\log k}{\sqrt{k}} \leq 48\sqrt{k}\log k$. For all $j \in N$,

$$\sum_{(i,C)\in\mathcal{C}, M_i(C)\ni j} x_{i,C} \leq f_j.$$

Adding up the above inequality for all j, we derive $\sum_{(i,C)\in\mathcal{C}} |M_i(C)|x_{i,C} \leq \sum_{j\in N} f_j$. We conclude:

$$\sum_{(i,C)\in\mathcal{C}} x_{i,C} \leq \frac{\sum_{j\in N} f_j}{\sqrt{k}\log^2 k/12} \leq 576/\log k = o(1). \tag{14}$$

Which completes the proof.

□

The above lemma shows that after step 3 of Algorithm 2 with probability $1-o(1)$, every person receives valuation at least $T/40\sqrt{k}\log^2 k$ from the goods left in the bundle she claims.

4.3. Resolving Conflicts. This is the last part of our analysis. Each person who was not saturated by the matching selected in step 2 of the main algorithm will claim some bundle consisting of small goods in step 3. In the previous section, we show that with high probability the value of unsaturated goods demanded by each person is large enough. The only issue is that there might be some conflicts in the bundles demanded by these people. As one can see, we deal with the issue in the last step of the algorithm.

The idea is to first define a fractional solution between the people who have not achieved a big good and the goods that have not been allocated yet. We define such a solution by dividing any such good equally between the people claiming it. In particular, if one good is claimed by only one person, she will get this good entirely. Also, if a good is not claimed by anyone, we allocate it to some person arbitrarily and will not use these goods in our analysis. After defining this fractional solution, we will use the additive approximation algorithm of [8] to allocate these goods. The rest of this section is devoted to analyze this last step of Algorithm 1.

Fix a good j. If $(i,j) \in \mathcal{F}$, the probability that i claims a bundle that includes j is $w_{i,j}$. Now, $\sum_i w_{i,j} \leq 1$, due to the constraints in configuration LP. Hence, applying Corollary 13 with probability at least $1 - k^{-\gamma/4}$ good j will be claimed by at most $\gamma \log k$ persons.

Fix a person i. By a Markov inequality argument with probability at least $1 - k^{-\gamma/8}$ no more than a fraction of $k^{-\gamma/8}$ of the goods in the bundle C that she has claimed are allocated to more than $\gamma \log k$ goods. Note that the same holds for the goods in $M_i(C)$ for which all the values are within a factor 2 of each other. Therefore the probability that person i still has a valuation at least $T/(80\sqrt{k}\log^2 k)$ among the goods in her selected bundle that no more than $\gamma \log k$ other persons have claimed, is at least $1 - k^{-\gamma/8}$. Choosing $\gamma = 9$, we conclude that with probability $1 - k^{-1/8}$ this event happens for all persons.

Now, we again use the idea of cycle elimination (see Lemma 1). We make a bipartite graph with people unsaturated by the matching on one side and unallocated goods on the other side. We connect person i to good j if j is in the bundle claimed by i in step 3. We put a weight $1/d_j$ on the edge (i,j) where d_j is the number of people who have claimed good j. This weight represents the fractional amount of good j that person i will receive if we divide j uniformly between all the people who want it.

Now, one can eliminate cycles in this graph in a way that the total value that each person receives according to the fractional allocation represented by the graph does not decrease. But by the Markov inequality argument before, we know that this value is at least $T/(80\sqrt{k}\log^2 k\gamma \log k) = T/720\sqrt{k}\log^3 k$ for all the people. Now, in order to resolve conflicts we make every tree in the graph rooted at some arbitrary vertex and give to each person all the goods corresponding to her children in the tree. In this way, each person will lose at most one good with value less than $T/1440\sqrt{k}\log^3 k$ and we get our main result.

THEOREM 17. *With probability* $1 - o(1)$ *our algorithm assigns each person a bundle with valuation at least* $\Omega(T/\sqrt{k}\log^3 k)$.

5. Further Discussion.

5.1. Extension. In our rounding mechanism, we can improve the guarantee for most of the people if we are allowed to leave a small fraction of the them unhappy. More precisely, let OPT be the optimal solution for the max-min fair problem. For any

$\alpha > 48 \log k$, we can allocate the goods to people so that everybody except $O(\frac{k}{\alpha^2 \log k})$, receive a bundle whose value is at least $\Omega(\frac{\text{OPT}}{\alpha \log^3 k})$.

In order to do this we need to redefine the notions of valid configuration, matching/flow bundles, and big/small goods by replacing $\sqrt{k}$ with α. We will use the same rounding method as in the Algorithm 1, but with respect to the new definition of matching and flow edges. The analysis of the algorithm remains the same except that $\sqrt{k}$ will be replaced by α in the extensions of Lemmas 14, 16 and Section 4.3. Here, a failing pair is similarly defined as all the pairs (i, C) so that at the end of Algorithm 2, i is not saturated by the matching and her valuation in the set of goods left in C is less than $T/40\alpha \log^2 k$. In the extension of Lemma 16 we can show that the expected number of failing pairs in the third step of our algorithm is of $O(\frac{k}{\alpha^2 \log k})$. More specifically, we will have $\sum_{j \in N} f_j = |N| O(\frac{\log k}{\alpha})$. Moreover, the size of $M_{i,C}$ for all failing pairs (i, C) will be $\Omega(\alpha \log^2 k)$ (since $R_i(C) = O(T/\alpha \log k)$). Thus, by rewriting (14) the expected number of failing bundles is at most

$$\sum_{(i,C) \in \mathcal{C}} x_{i,C} \leq \frac{\sum_{j \in N} f_j}{\min_{(i,C) \in \mathcal{C}} M_{i,C}} = \frac{|N| O(\frac{\log k}{\alpha})}{\alpha \log^2 k} = O(\frac{k}{\alpha^2 \log k}).$$

Therefore, by Markov inequality, with probability at least $1/2$, the actual number of failing pairs will not be more than twice this number. Hence, everybody, except at most $O(\frac{k}{\alpha^2 \log k})$ people receive a bundle of value at least $\Omega(\frac{\text{OPT}}{\alpha \log^3 k})$ after resolving the conflicts (and losing another factor of $\log k$).

5.2. Maximum Entropy. The principle of *Maximum Entropy*, first introduced in Jaynes [17], is a well-known method to derive the least biased distribution that encodes specific given information. We start this section by a quick review on the principle of maximum entropy applied on the problem of sampling random matchings in bipartite graph. For more detailed description one might refer to [18].

Consider the following problem:

> For a given doubly stochastic matrix $A_{n \times n}$, we want to derive a distribution D over permutations so that the probability that each single entry (i, j) appears in a permutation sampled from D is A_{ij}. In other words, we want to set the values of p_i such that
>
> $$A = \sum_i p_i M_i, \tag{15}$$
>
> where the sum is taken over all $n \times n$ permutation matrices. Note that it can be interpreted as the problem of sampling a random matching in a bipartite graph with given marginal probabilities for the edges. We will use this two notions interchangeably.

Suppose that we want to find such a distribution which is least biased or equivalently, does not add any information other than the constraints already imposed by equation (15). The principle of maximum entropy states that p_i's should be derived

from the following optimization problem:

$$\begin{aligned} \textbf{maximize} \sum_i -p_i \log p_i \\ \textbf{subject to: } \forall e: \sum_{M_i \ni e} p_i = w_e, \\ \sum_i p_i = 1, \\ \forall i: \ 0 \le p_i \le 1. \end{aligned}$$

Note that the objective function is strictly concave. Therefore, there exists a unique local maximum which is also global maximum. The KKT condition for the program states that in the optimal point p^*, one can find μ_e, λ, α_i and β_i such that:

$$\forall i: -1 - \log p_i^* + \sum_{e \in M_i} \mu_e + \lambda - \alpha_i + \beta_i = 0, \tag{16}$$

$$\forall e: \sum_{M_i \ni e} p_i^* = w_e, \tag{17}$$

$$\sum_i p_i^* = 1, \tag{18}$$

$$\forall i: 0 \le p_i^* \le 1, \alpha_i p_i^* = 0, \beta_i (1 - p_i^*) = 0. \tag{19}$$

Condition (19) simply says that if the probability of selecting a matching M_i is positive, then $\alpha_i = 0$. Also, one can see that p_i^* can be 1 at most for one matching M_i, in which case the problem becomes trivial. Therefore, we can assume $\beta_i = 0$ for all i.

Now, rearranging condition (16) we have:

$$p_i^* = \frac{e^{\sum_{e \in M_i} \mu_e}}{e^{1+\lambda}}. \tag{20}$$

It is clear that p_i^*'s are from an exponential family. For simplicity let us define new variable $\eta_e = e^{\mu_e}$. Hence, in the optimum point there are η_e's such that

$$p_i^* \propto \prod_{e \in M_i} \eta_e,$$

and the scaling factor Z, which is simply $e^{1+\lambda}$ in the denominator of equation (20), comes from the following:

$$Z = \sum_i \prod_{e \in M_i} \eta_e.$$

Now, we go back to the second step of our algorithm in which we selected a random matching in our graph. We saw that without loss of generality, we can assume that $\mathcal{M}$ is a forest. Then we gave an iterative algorithm for rounding the edges of $\mathcal{M}$ to get a matching. Define a distribution D on the possible matchings of $\mathcal{M}$ feasible if and only if for every vertex v, the probability that it is saturated by a matching in D is m_v.

THEOREM 18. *The distribution over matchings of $\mathcal{M}$ imposed by the output of Algorithm 2 has the maximum entropy among all feasible distributions.*

Proof. As we saw in the above discussion D^* , the distribution with the highest entropy, belongs to an "exponential family". In other words, it is possible to find $\eta : E(G) \mapsto \mathbb{R}$ such that the probability of a certain matching M in D^* is proportional to $\Pi_{e\in M}\eta(e)$. It is easy to show that in a distribution like that the events corresponding to two different trees in the forest are independent. Now, consider a path $e_0, e_1, \cdots, e_k$ in the tree . It follows from the Bayes' rule that

$$\begin{aligned}\Pr[e_0 \in M | e_k \notin M] = \Pr[e_0 \in M | e_1 \notin M, e_k \notin M] \Pr[e_1 \notin M | e_k \notin M] + \\ \Pr[e_0 \in M | e_1 \in M, e_k \notin M] \Pr[e_1 \in M | e_k \notin M]\end{aligned}$$

Note that since e_1 lies between e_k and e_0, then conditioned on $e_1 \notin M$, the edges e_0 and e_k fall in two different forest which make the probability of their appearance independent. Therefore,

$$\Pr[e_0 \in M | e_1 \notin M, e_k \notin M] = \Pr[e_0 \in M | e_1 \notin M].$$

Moreover, e_0 and e_1 cannot be in the matching at the same time. Hence,

$$\Pr[e_0 \in M | e_1 \in M, e_k \notin M] = 0.$$

Thus, we conclude

$$\Pr[e_0 \in M | e_k \notin M] = \Pr[e_0 \in M | e_1 \notin M] \Pr[e_1 \notin M | e_k \notin M].$$

Now, we can prove by a simple induction on k, that if a distribution D satisfies this property, then

$$\Pr[e_0 \in M | e_k \notin M] = x_{e_0} + (-1)^{k-1} \frac{x_{e_1} \cdots x_{e_k}}{(1 - x_{e_1}) \cdots (1 - x_{e_k})} x_{e_0}.$$

To see the above, we note that the base of induction (k = 1) trivially holds. For $k \geq 2$ we have

$$\begin{aligned}\Pr[e_0 \in M | e_k \notin M] &= \Pr[e_0 \in M | e_1 \notin M] \Pr[e_1 \notin M | e_k \notin M] \\ &= \frac{x_{e_0}}{1 - x_{e_1}} [1 - x_{e_1} - (-1)^{k-2} \frac{x_{e_2} \cdots x_{e_k}}{(1 - x_{e_2}) \cdots (1 - x_{e_k})} x_{e_1}] \\ &= x_{e_0} + (-1)^{k-1} \frac{x_{e_1} \cdots x_{e_k}}{(1 - x_{e_1}) \cdots (1 - x_{e_k})} x_{e_0}.\end{aligned}$$

where x_e is the probability that e is picked in a matching M selected with respect to D. This is exactly the update rule of our algorithm. Thus, our algorithm yields the maximum entropy probability distribution. □

As an immediate result of the above discussion, we can implement Algorithm 2 differently. We first solve the convex program (16) to find out the desired probability distribution over matchings. In fact we derive the values of η and Z. Then we can easily sample a random matching from the probability distribution imposed by (20).

REMARK 2. *Theorem 18 immediately gives a proof for Lemma 4. Note that entropy is a strictly concave function and hence, the maximum entropy distribution is unique. Also, by Theorem 18 we know that our algorithm, independent of the order of vertices that it chooses, produces a distribution that maximizes the entropy with*

respect to the initial marginals. Hence, the outcome of our algorithm is a unique distribution which is independent from the order of vertices.

REMARK 3. *Later, we extended our rounding method based on sampling from a maximum entropy distribution to other combinatorial structures, including balanced matroids (see [12] for the definition of balanced matroids). In particular, our rounding method, being used to sample trees, played a crucial part in improving the approximation factor for the asymmetric traveling salesman problem (ATSP) to $O(\log\log n/\log\log\log n)$ [4], providing the first asymptotic improvement in the approximation factor 28 years after the work of Frieze et al [15].*

5.3. Open Directions. The most challenging question is to narrow the gap between the approximation ratio of $\Omega(\frac{1}{\sqrt{k}\log^3 k})$ presented here and the factor $\frac{1}{2}$ hardness result of [8]. To obtain a considerable improvement for the approximation ratio one needs to come up with something more than the configuration LP as we already know that its integrality gap is $\sqrt{k}$. Also, improving the current results for some special cases of the problem would be interesting. As an example, one promising way to improve the result of [6] for the restricted assignment problem is to establish a polynomial upper bound for the convergence time of the local search proposed by [3].

Acknowledgement: We would like to thank Uriel Feige, Soheil Mohajer, Nikhil Bansal, and Nicole Immorlica for insightful discussions and useful comments on an earlier version of this paper. We would also like to thank Mohsen Bayati and Andrea Montanari for useful discussions on maximum-entropy distributions. We also thank two anonymous referees for their useful comments to improve the writing of the paper.

REFERENCES

[1] N. ALON AND J. H. SPENCER, *The Probabilistic Method*, second edition, John Wiley & Sons, Inc. 2000.

[2] S. ARORA, A. FRIEZE, AND H. KAPLAN, *A New Rounding Procedure for the Assignment Problem with Applications to Dense Graph Arrangement Problems*, in Mathematical Programming, A 92:1-36, 2002.

[3] A. ASADPOUR, U. FEIGE, AND A. SABERI, *On Max-Min Fair Allocations and Hypergraph Matchings*, in the Proceedings of APPROX-RANDOM, 2008:10-20.

[4] A. ASADPOUR, M. GOEMANS, A. MADRY, S. OVEIS GHARAN, A. SABERI, *An $O(\log n/\log\log n)$-approximation Algorithm for the Asymmetric Traveling Salesman Problem*, to appear in ACM-SIAM Symposium on Discrete algorithms (SODA), 2010.

[5] A. ASADPOUR AND A. SABERI, *An Approximation Algorithm for Max-Min Fair Allocation of Indivisible Goods*, in the Proceedings of the ACM Symposium on Theory of Computing (STOC), 2007:114-121.

[6] N. BANSAL AND M. SVIRIDENKO, *The Santa Claus problem*, in the Proceedings of the ACM Symposium on Theory of Computing (STOC), 2006:31-40.

[7] M. BATENI, M. CHARIKAR, AND V. GURUSWAMI, *MaxMin Allocation via Degree Lower-bounded Arborescences*, in the Proceedings of the ACM Symposium on Theory of Computing (STOC), 2009:543-552.

[8] I. BEZAKOVA AND V. DANI, *Allocating Indivisible Goods*, in SIGecom Exchanges 5(3), 2005:11-18.

[9] S. J. BRAMS AND A. D. TAYLOR, *Fair Division: from Cake Cutting to Dispute Resolution*, Cambridge University Press, 1996.

[10] D. CHAKRABARTY, J. CHUZHOY, AND S. KHANNA, *On Allocating Goods to Maximize Fairness*, to appear in IEEE Symposium on Foundations of Computer Science (FOCS), 2009.

[11] F. CHUNG AND L. LU, *Coupling Online and Offline Analyses for Random Power-law Graphs*, in Internet Mathematics, 1(4), 2004.

[12] T. Feder and M. Mihail, *Balanced Matroids*, in the Proceedings of the ACM Symposium on Theory of Computing (STOC), 1992:26-38.
[13] U. Feige, *On Allocations that Maximize Fairness*, in the Proceedings of ACM-SIAM Symposium on Discrete algorithms (SODA), 2008: 287-293.
[14] L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko, *Tight Approximation Algorithms for Maximum General Assignment Problems* in the Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA), 2006:611-620.
[15] A. M. Frieze, G. Galbiati, and F. Maffioli, *On the Worst-case Performance of Some Algorithms for the Asymmetric Traveling Salesman Problem*, Networks, Volume 12, 1982:23-39.
[16] D. Golovin, *Max-min Fair Allocation of Indivisible Goods*, Technical Report, Carnegie Mellon University, CMU-CS-05-144, 2005.
[17] E.T. Jaynes, *Information Theory and Statistical Mechanics*, Physical Review, Volume 106.4, 1957:620-630.
[18] E.T. Jaynes, *Probability Theory: The Logic of Science*, Cambridge University Press.
[19] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis, *How Easy is Local Search?*, Journal of Computer and System Sciences, 37, 1988:79-100.
[20] S. Khot, A. K. Ponnuswami, *Approximation Algorithms for the Max-Min Allocation Problem*, APPROX-RANDOM 2007: 204-217.
[21] J. Kleinberg, Y. Rabani, E. Tardos, *Fairness in Routing and Load Balancing*, Journal of Computer and System Sciences, 63(1), 2001: 2-20.
[22] A. Kumar, J. Kleinberg, *Fairness Measures for Resource Allocation*, SIAM Journal on Computing 36(3), 2006:657-680.
[23] J.K. Lenstra, D.B. Shmoys, and E. Tardos, *Approximation Algorithms for Scheduling Unrelated Parallel Machines*, Mathematical Programming, 1990, 46: 259-271.
[24] R. Lipton, E. Markakis, E. Mossel, and A. Saberi, *On Approximately Fair Allocations for Indivisible Goods*, in the Proceedings of ACM Conference on Electronic Commerce (EC), 2004: 125-131.
[25] J. M. Robertson and W. A. Webb, *Cake-Cutting Algorithms: Be Fair If You Can*, AK Peters, 1998.
[26] H. Steinhaus, *The Problem of Fair Division*, Econometrica, 1948, 16:101-104.
[27] V.V. Vazirani, *Approximation Algorithms*, Springer-Verlag, Spring 2001.

Appendix A. An Instance of Max-Min Fair Problem with Large Integrality Gap. Figure 4 shows an instance of the problem which admits an integrality gap of $O(\frac{1}{\sqrt{k}})$. This instance was first introduced in [6]. A close inspection of it reveals that rounding the matching edges need to be done in a very careful manner.

Let $T > 1$ be an integer number. The black and void circles correspond to the goods and people, respectively. There are $T^2 + T$ people and $T^2 + 2T - 1$ goods. The value of solid and dashed edges are T and 1, respectively. The configuration LP is feasible for value T. Such a fractional solution is given here. The fractional numbers inside the parentheses between layers are the fraction of goods that all edges between those layers carry in the fractional solution. The thicker edges are the ones that are selected to be in the integral solution.

Note that the edges should be rounded carefully, since at least one of the people in layer 2 will not receive any good from layer 1. That person will need to get some goods from its corresponding bundle in layer 3. If she gets more than one goods from layer 3, then at least one of the people in layer 4 will receive nothing. So she should get exactly one good from layer 3. In that case, the people in layer 4 need to gather all their goods in layer 3 (except exactly the one that is claimed by the person in layer 2) and their single good in layer 5.

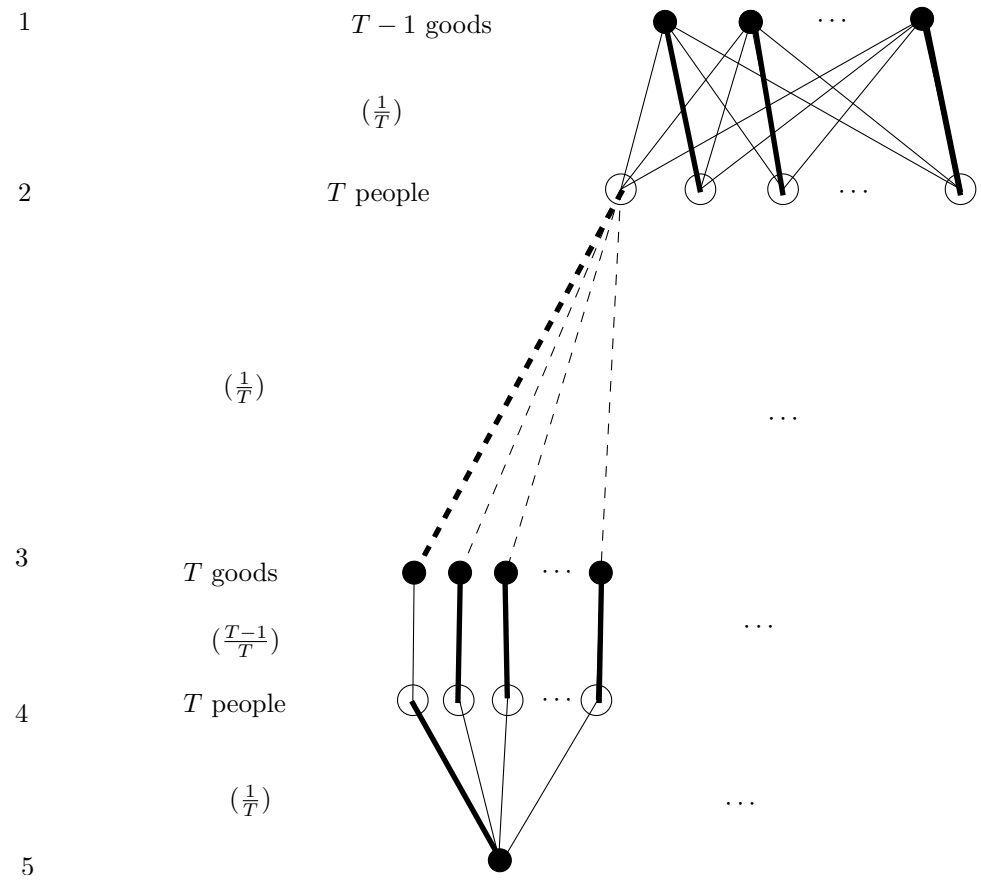

FIG. 4. *An instance of max-min fair allocation problem from [6].*