Data Mining and Knowledge Discovery (2020) 34:1291–1335
https://doi.org/10.1007/s10618-020-00675-y

Fair-by-design matching

David García-Soriano[1] · Francesco Bonchi[1,2]

Received: 21 January 2019 / Accepted: 13 January 2020 / Published online: 4 February 2020

Abstract

Matching algorithms are used routinely to match donors to recipients for solid organs transplantation, for the assignment of medical residents to hospitals, record linkage in databases, scheduling jobs on machines, network switching, online advertising, and image recognition, among others. Although many optimal solutions may exist to a given matching problem, when the elements that shall or not be included in a solution correspond to individuals, it becomes of paramount importance that the solution is selected fairly. In this paper we study individual fairness in matching problems. Given that many maximum matchings may exist, each one satisfying a different set of individuals, the only way to guarantee fairness is through randomization. Hence we introduce the *distributional maxmin fairness* framework which provides, for any given input instance, the strongest guarantee possible simultaneously for all individuals in terms of satisfaction probability (the probability of being matched in the solution). Specifically, a probability distribution over feasible solutions is *maxmin-fair* if it is not possible to improve the satisfaction probability of any individual without decreasing it for some other individual which is no better off. Our main contribution is a polynomial-time algorithm building on techniques from minimum cuts, and edge-coloring algorithms for regular bipartite graphs, and transversal theory. In the special case of bipartite matching, our algorithm runs in $O((|V|^2 + |E||V|^{2/3}) \cdot (\log |V|)^2)$ expected time. An experimental evaluation of our fair-matching algorithm shows its ability to scale to graphs with tens of millions of vertices and hundreds of millions of edges, taking only a few minutes on a simple architecture. To the best of our knowledge, this yields the first large-scale implementation of the egalitarian mechanism of Bogomolnaia and Moulin (Econometrica 72(1):257–279, 2004). Our analysis confirms that our method provides stronger satisfaction probability guarantees than non-trivial baselines.

Keywords Algorithmic bias · Fairness · Matching · Combinatorial optimization

Responsible editor: Karsten Borgwardt, Po-Ling Loh, Evimaria Terzi, Antti Ukkonen.

✉ David García-Soriano
d.garcia.soriano@isi.it

Extended author information available on the last page of the article

1 Introduction

Decision-making tools relying on data and quantitative measures have become pervasive in application domains such as education and employment, finance, search and recommendation, policy making, and criminal justice. Awareness and concern about the risks of unfair automated decisions is quickly rising, as it has been argued that decisions informed by data analysis could have inadvertent discriminatory effects due to potential bias existing in the data or encoded in automated decisions. Several reports (Big data 2014, 2016) call for algorithms that are *fair by design* and identify *poorly designed matching systems* as one of the main flaws of algorithmic decision-making. The way to tackle the ensuing ethical and societal issues has garnered the attention of the research community (Dwork 2017). However, despite the fact that matching mechanisms lie at the basis of many automated decision systems, the bulk of the research in the area of algorithmic bias and fairness has mainly focused on avoiding discrimination against a sensitive attribute (i.e., a protected social group) in supervised machine learning (Kearns 2017).

Our work departs from this literature in three main directions: (1) we focus on *individual fairness* (as opposed to group-level fairness); (2) we focus on *bias stemming from the algorithm design itself*, rather than the bias existing in the input data; (3) instead of supervised learning we focus on *matching problems*, where the solution may not be unique and individuals correspond to elements to be included in the solution.

In this setting, the satisfaction (utility) function of each individual is based on whether the individual has been selected or not for inclusion. At the very least, two individuals satisfying all relevant criteria equally well (e.g., having the same skill set) should have, in principle, the same expected utility; moreover, individuals having a wider or more unique skill set (covering relevant criteria that others can't cover), should reasonably be rewarded with higher expected utility. This is often not the case as algorithms may be *biased by design*: bias may stem from something as petty as the order in which the algorithm chooses to process the list of candidates in its main loop (e.g., by irrelevant attributes such as alphabetical order or application date), or details about the internal workings of the algorithm. The prototypical example of a "biased by design" algorithm (in a rather extreme way) arises in the context of *stable matching* (a different problem from the one considered in this paper): the Gale–Shapley algorithm (Gale and Shapley 1962) produces a solution which is always the best for every man and the worst for every woman, among all feasible solutions, despite the existence of another solution which lies provably "in the middle" for every man and woman (Teo and Sethuraman 1998).

Algorithmic bias and randomization Consider a job-search setting where we have a certain number of positions and applicants. Assume that each applicant has a binary fitting for each of the positions (either she is fit for the job or not) and a binary satisfaction function (either she is selected or not). This can be modeled as a matching problem in a bipartite graph. Unless a matching covering simultaneously all applicants exists, some of them will have to be left out. An unselected applicant could notice that there are other matchings (even maximum-size matchings) satisfying her. However, any deterministic algorithm is programmed to pick a specific one which may not include her: she might rightfully deem this unfair.

Unlike the Gale–Shapley algorithm, whose bias can be simply characterized by a theorem, for the problems we consider in this paper it may be hard to tell in advance which particular individuals a given algorithm favours. However, the fact that the bias is not easy to pinpoint does not mean it does not exist, just that we do not know what it is.

Since no single candidate solution satisfying all individuals at the same time can exist in general, *we turn our attention to randomized algorithms*, which make random choices to pick from among several valid solutions.

In our job-search example, imagine there is a single open position and n applicants fit for it. Intuitively, all applicants are "equally qualified" in this case and the fairest solution would choose one of them uniformly at random, giving each applicant a guaranteed satisfaction (matching) probability of $1/n$. However, as the graph between applicants and jobs grows more complex, it becomes unclear how to proceed, or what properties one should demand of a fair distribution of solutions. Our next example illustrates why requiring exactly the same satisfaction probability for all individuals would not make for a good definition.

Example 1 (Satisfaction probability) Consider the problem of finding a matching on the bipartite graph of Fig. 1 between people (on the left) and jobs (on the right). Let $\mathcal{U} = \{a_0, a_1, a_2, a_3\}$ and let $\mathcal{S}$ denote the set of all possible matchings. An individual $u \in \mathcal{U}$ is satisfied by a solution $S \in \mathcal{S}$ iff it is matched in S (i.e., she is selected for the job). Consider the distribution D assigning probability $\frac{1}{3}$ to each of the following solutions: $M_1 = \{(a_0, b_0), (a_1, b_1), (a_2, b_2)\}$, $M_2 = \{(a_0, b_0), (a_1, b_1), (a_3, b_2)\}$, $M_3 = \{(a_2, b_2), (a_3, b_1)\}$ and zero probability to all the other matchings.

The satisfaction probability of each individual under distribution D is exactly the same, namely $\frac{2}{3}$. While D might naively look "fair", notice that the job b_0 is left unassigned in M_3, despite the existence of a fitting candidate occasionally left unemployed (a_0). This artificially restricts the satisfaction probability of a_0. Observe that, for any matching covering a subset $T \subseteq \{a_1, a_2, a_3\}$, there is another matching covering $T \cup \{a_0\}$. So a_0 can always be satisfied without impacting anyone else's chances,

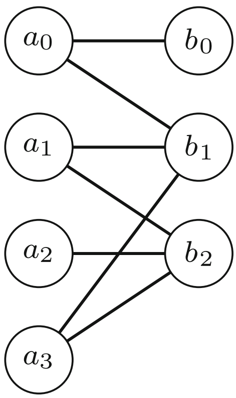

Fig. 1 An example bipartite graph between people (on the left) and jobs (on the right)

hence any reasonable solution should match a_0 with probability 1. Other applicants will have lower satisfaction probability though (as no matching can satisfy all of a_1, a_2 and a_3 at the same time).

The insight from Example 1 leads us to the key definition of our work. Our aim is to provide, on any given input instance, the strongest guarantee possible for all individuals, in terms of satisfaction probability. We thus introduce the *distributional maxmin fairness* framework. Informally, a distribution over matchings is *maxmin-fair* if it is impossible to improve the satisfaction probability of any individual without decreasing it for some other individual which is no better off (see Sect. 2 for a formal definition).

Example 2 (Maxmin-fair distribution) Consider Example 1 again. A distribution assigning non-zero probability to a solution not covering a_0 (such as M_3) cannot be maxmin-fair, as otherwise one can increase the satisfaction probability of a_0 without detriment to anyone else. On the other hand, notice that $\{a_1, a_2, a_3\}$ have only two neighbors $\{b_1, b_2\}$, making it impossible to guarantee satisfaction probability $\geq \frac{2}{3}$ for a_1, a_2 and a_3 at the same time. This graph has four maximum matchings: M_1 and M_2 from Example 1, $M_4 = \{(a_0, b_0), (a_1, b_2), (a_3, b_1)\}$, and $M_5 = \{(a_0, b_0), (a_2, b_2), (a_3, b_1)\}$. The distribution F_1 choosing from among M_1, M_2, M_4 and M_5 with probability $\frac{1}{3}, \frac{1}{6}, \frac{1}{6}, \frac{1}{3}$, respectively, is maxmin-fair. The satisfaction probabilities of a_0, a_1, a_2 and a_3 are then $1, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}$. Any attempt to match, say, a_1 with probability $> \frac{2}{3}$ will necessarily result in satisfaction probability $< \frac{2}{3}$ for a_2 or a_3. Another maxmin-fair distribution is, e.g., the distribution F_2 choosing uniformly at random from among M_1, M_2 and M_5.

Overview of our contributions The contributions of this paper can be summarized as follows:

- We introduce and characterize the *distributional maxmin-fairness* framework providing, on any given problem instance, the strongest guarantee possible for all individuals, in terms of satisfaction probability (Sect. 2). While in this paper, for sake of simplicity of presentation, we focus on matching problems, our definition applies to a wider variety of problems (such as those listed in Example 5).
- In Sect. 3, we show that when the structure of valid solutions forms a matroid (which is the case for matchings), maxmin-fairness minimizes the largest inequality gap in satisfaction probabilities between all pairs of individuals, among all Pareto-efficient distributions (Theorem 2). We also observe that for such problems the "price of fairness" is zero: maxmin fairness is attainable at no cost in solution size.
- We give a characterization of the "degree of fairness" attainable in any bipartite matching instance (Corollary 2) and any matroid problem instance (Theorem 15), generalizing the classical marriage theorem due to Hall (1935).
- We apply our framework to matching problems in bipartite graphs (Sects. 4 and 5), leading to our main contribution: an exact algorithm for *maxmin-fair bipartite matching* with running time $O((|V|^2 + |E||V|^{2/3}) \cdot (\log |V|)^2)$ (Theorem 7). We also obtain a polynomial time maxmin-fair algorithm for matching in general graphs by a reduction to the aforementioned bipartite case (Theorem 12).

- We discuss how to achieve full transparency for real-world deployment of our framework (Sect. 7). The discussion leads to the problem of producing a maxmin-fair distribution with small support, for which we offer an approach making small modifications to our algorithm.
- Our experiments (Sect. 8) show that our algorithm performs faster in practice than its theoretical running time and scales to graphs with tens of millions of vertices and hundreds of millions of edges, taking only a few minutes on a simple architecture. Our analysis confirms that our method provides stronger satisfaction probability guarantees than non-trivial baselines.

2 Problem definition

In this section we provide the key definition of our *distributional maxmin-fairness* framework, considering a very general *search problem* instance $\mathcal{I} = (\mathcal{U}, \mathcal{S})$ defined over a finite set of individuals $\mathcal{U}$ and where $\mathcal{S} \neq \emptyset$ denotes the set of feasible solutions for the problem instance $\mathcal{I}$. (For example, instance $\mathcal{I}$ could represent a bipartite graph between jobs and a set $\mathcal{U}$ of applicants, and $\mathcal{S}$ the set of all matchings.) We assume that for every solution $S \in \mathcal{S}$, each individual $u \in \mathcal{U}$ is either fully satisfied or fully dissatisfied, and this is the only property of the solution we are concerned with. Thus, for the sake of simplicity, we will identify each solution in $\mathcal{S}$ with the subset of users satisfied by it, so $\mathcal{S} \subseteq 2^{\mathcal{U}}$. Note that $\mathcal{S}$ is defined implicitly by the structure of the problem, and not explicitly encoded in the input.

Given $\mathcal{I}$, our problem is to return an element of $\mathcal{S}$ while providing a fairness guarantee to all individuals in $\mathcal{U}$. Since in general no single candidate solution satisfying all $u \in \mathcal{U}$ at the same time exists ($\mathcal{U} \notin S$), we seek a *randomized algorithm* $\mathcal{A}$ that, for any given problem instance $\mathcal{I}$, always halts and selects one solution $\mathcal{A}(\mathcal{I})$ from $\mathcal{S}$. Thus $\mathcal{A}$ induces a probability distribution D over $\mathcal{S}$: $\Pr_D[S] = \Pr[\mathcal{A}(\mathcal{I}) = S]$ for each $S \in \mathcal{S}$. The *satisfaction probability* of each individual $u \in \mathcal{U}$ under D is defined by $D[u] = \Pr_{S \sim D}[u \in S]$.

Based on the insight from Example 1, we next provide the key definition of our work. Informally, a distribution over solutions is *maxmin-fair* if it is impossible to improve the satisfaction probability of any individual without decreasing it for some other individual which is no better off.

Definition 1 (Maxmin-fairness) A distribution F over $\mathcal{S}$ is *maxmin-fair* for $\mathcal{U}$ if for all distributions D over $\mathcal{S}$ and all $u \in \mathcal{U}$,

$$D[u] > F[u] \implies \exists v \in \mathcal{U} \mid D[v] < F[v] \leq F[u]. \tag{1}$$

Similarly, a randomized algorithm is maxmin-fair if it induces a maxmin-fair distribution.

Finding a maxmin-fair distribution involves solving a continuous optimization problem over (infinitely many) distributions over the set $\mathcal{S}$ of valid solutions (which is commonly exponential in size). The challenge we face is thus how to design an efficient randomized algorithm inducing a maxmin-fair distribution.

Table 1 Summary of notation

Symbol	Meaning
$G = (V, E)$	Undirected, unweighted graph with vertex set V and edge set E
$\mathcal{U}$	Set of users; $\mathcal{U} \subseteq V$ for matchings
$\mathcal{S} \subseteq 2^{\mathcal{U}}$	Collection of feasible solutions (possible subsets of satisfied users)
$\Gamma_G(A)$	Set of neighbours of $A \subseteq V$ in G
$V = L\dot{\cup}R$	Bipartition of the vertex set V of a bipartite graph
$\rho(A)$	(For graphs) size of the largest matchable subset of $A \subseteq V$
$\rho : 2^L \to \mathbb{N}$	(For matroids) rank function of a matroid with ground set L
D	Distribution of subsets of $\mathcal{S}$
$D[v]$	Satisfaction probability of user u under distribution D
$D^{\uparrow}$	Vector of satisfaction probabilities of D in increasing order
$D^{\downarrow}$	Vector of satisfaction probabilities of D in decreasing order
$\succ$	Lexicographical order of vectors
$\pi(G)$	Minimum satisfaction probability of a maxmin-fair distribution for G
$\Pi(G)$	Maximum satisfaction probability of a maxmin-fair distribution for G
x_{uv}	Probability of u being matched to v in a fixed maxmin-fair distribution
$B_1, \ldots, B_k$	Fair decomposition of L into blocks
$(F(X))_{X\sim D}$	Distribution of random variable $F(X)$ when X is drawn from D
$G\|_A$	Subgraph of G induced by $A \cup \Gamma(A)$
$M\|_A$	Restriction of matroid M to the set A
G/A	Subgraph of G induced by $(L \cup R)\setminus(A \cup \Gamma(A))$
M/A	Contraction of matroid M to the set A

Problem 1 For a given search problem, design a randomized algorithm $\mathcal{A}$ which always terminates and such that, for each instance $\mathcal{I} = (\mathcal{U}, \mathcal{S})$, the distribution of $\mathcal{A}(\mathcal{I})$ is maxmin-fair for $\mathcal{U}$ over $\mathcal{S}$.

While our definition applies to a wider variety of search problems, in this paper, for sake of simplicity of presentation, we solve Problem 1 in the case where the search problem is a matching problem in a graph. Let us specify what the sets $\mathcal{U}$ of users and $\mathcal{S}$ of solutions are in this case (Table 1 below summarizes the notation used throughout the paper).

Let $G = (V, E)$ be an unweighted simple graph. A *matching* in G is a set of vertex-disjoint edges of G. A *maximum* matching is a matching of largest size. The matching M *covers* a vertex $v \in V$ if v is incident to some edge in M. A set $S \subseteq V$ is *matchable* if there is a matching of G covering all of S. For $S \subseteq V$, define $\rho_G(S)$ as the size of the largest matchable subset of S; then $\rho_G(V)$ is the size of the maximum matching of G. Denote by $\Gamma_G(S)$ the set of neighbours of S in G. We will drop the G subscript when no confusion may arise.

In the *fair matching* problem, the input is a graph $G = (V, E)$ and a set $\mathcal{U} \subseteq V$ of users. Following our assumption of binary satisfaction, user $u \in \mathcal{U}$ is *satisfied* by a matching M if u is covered by M. The set $\mathcal{S}$ of valid solutions is the set of matchable subsets of $\mathcal{U}$. The set $\mathcal{S}$ is not part of the input given to the algorithm, but implicitly defined by G and $\mathcal{U}$.

While the results provided in Sect. 3 hold for fair-matching on a general graph, the algorithms developed in Sects. 4 and 5 are for the interesting special case of *one-sided fair bipartite matching* problem, i.e., where G is bipartite (with bipartition $V = L \dot{\cup} R$) and the set of users is given by $\mathcal{U} = L$. By solving the one-sided fair bipartite matching problem, we also obtain a polynomial time maxmin-fair algorithm for matching in general graphs by means of a reduction to the bipartite case (see Theorem 12).

3 Fairness and social inequality

In this section we present several properties of maxmin-fair distributions. These are of independent interest as they provide alternative definitions of maxmin-fairness (Theorems 1 and 2) which are arguably just as natural as Definition 1; moreover, the latter offers insights into the inequality distribution properties of maxmin-fairness. Some results are only stated here; their proofs may be found in “Appendix A”.

3.1 Basic properties of maxmin-fair distributions

An important preliminary observation is that maxmin-fair distributions are unique as far as satisfaction probabilities go, even though several ways may exist to achieve the optimal satisfaction probabilities.

Lemma 1 *Let F and D be two maxmin-fair distributions. Then $F[u] = D[u]$ for all $u \in \mathcal{U}$.*

Example 3 In Example 2 we gave two maxmin-fair distributions, F_1 and F_2, which are obtained by combining maximum matchings in different ways, but both satisfy $F_1[a_0] = F_2[a_0] = 1$ and $F_1[x] = F_2[x] = \frac{2}{3}$ for $x \in \{a_1, a_2, a_3\}$.

Given a distribution D over $\mathcal{S}$, write $D \uparrow = (\lambda_1, \dots, \lambda_n)$ for the vector of satisfaction probabilities $(D[u])_{u \in \mathcal{U}}$ sorted in increasing order. Let $\succ$ denote the lexicographical order of vectors: $(v_1, \dots, v_n) \succ (w_1, \dots, w_n)$ iff there is some index $i \in [n]$ such that $v_j = w_j$ for all $j < i$ and $v_i > w_i$ (the relations $\succeq$, $\prec$ and $\preceq$ are defined similarly). The following holds.

Theorem 1 *A distribution F is maxmin-fair if and only if $F \uparrow \succeq D \uparrow$ for all distributions D over $\mathcal{S}$.*

In other words, a maxmin-fair distribution maximizes the smallest satisfaction probability; subject to that, it maximizes the second-smallest satisfaction probability, and so on.

Example 4 In Examples 1 and 2 we have $F_1 \uparrow = F_2 \uparrow = (1, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}) \succ D \uparrow = (\frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3})$. As $D \uparrow$ is not lexicographically maximal, it cannot be maxmin-fair; whereas $F_1 \uparrow$ can be shown to be lexicographically maximal, and hence F_1 is maxmin-fair.

An important observation is that a maxmin-fair distribution always exists for any search problem instance with a feasible solution:

Corollary 1 *Given a search problem instance $\mathcal{I} = (\mathcal{U}, \mathcal{S})$, a maxmin-fair distribution always exists.*

Proof The probability vectors defining distributions over $\mathcal{S}$ form a non-empty compact set, and the mapping from such vectors to their corresponding sorted satisfaction vectors is continuous, so the claim follows from Weirstrass theorem. □

3.2 Matroid problems

Theorem 1 above provides a definition of maxmin-fairness alternative to Definition 1. At the end of this section (Theorem 2) we provide a second alternative definition characterizing the inequality properties of the distribution of satisfaction probabilities, i.e., the differences between the satisfaction of the least and most satisfied individuals. It turns out that for a large class of problems, this difference is minimized in a maxmin-fair distribution, making the maxmin-fair distribution the most equitable. However this does not hold in general for all search problems.

To be able to state the class of problems for which Theorem 2 holds, we need to review the concept of *matroids*. Many search and optimization problems can be formulated in terms of matroids; they also provide a convenient framework to state and simplify the proofs of some of our results.

Definition 2 (Matroid problem) Let L be a finite set. A matroid with ground set L is a non-empty collection M of subsets of L satisfying the following two properties: (1) if $A \in M$ and $B \subseteq A$, then $B \in M$; (2) for any $X \subseteq L$, all maximal subsets of X (with respect to set inclusion) belonging to M have the same size.

A search problem is a *matroid problem* if for any instance $\mathcal{I} = (\mathcal{U}, \mathcal{S})$, the set $\mathcal{S}$ is a matroid. The elements of a matroid M are called *independent sets*. The maximal elements of M are called *bases*. All bases have the same size. The *rank function* of M is $\rho_M(S) = \max\{|X| \mid X \subseteq S, X \in M\}$.

Example 5 The following are matroids [see Lawler (1976)]:

- The collection of sets of matchable vertices in a graph (Lovász and Plummer 2009). This well-known result follows from a theorem of Berge (1957) that we may extend any matchable set of *vertices* to a matchable set of maximum size. By contrast, the collections of sets of *edges* forming a matching is *not* a matroid.
- The collection of sets of vertices in a directed graph for which edge-disjoint paths from another single specified vertex exist.
- The collection of linearly independent sets of vectors over a finite vector space.
- The collection of forests (acyclic sets of edges) in a graph.

The search problems corresponding to finding any of the above are matroid problems.

Notice that any set X appearing with non-zero probability in a maxmin-fair distribution must be maximum in size; otherwise, by property (2) in Definition 2, X is not maximal so we could replace X with some strict superset $Y \supsetneq X$, which can only increase the satisfaction probability of every $u \in L$. It is in this sense that the “price of fairness” is zero for matroid problems: the support of a maxmin-fair distribution consists only of solutions of maximum size, so it is never necessary to trade fairness for solution size. In particular this holds for matching problems as well.

3.3 Minmax-fairness

By definition, maxmin-fair distributions give the highest possible satisfaction probabilities to the worst-off individuals. To investigate the inequality properties of these, we introduce a dual notion of minmax-fair distributions, which by contrast give the lowest possible satisfaction probabilities to the best-off individuals. It turns out that for matroid problems both notions coincide, provided that we exclude Pareto-inefficient distributions.

Definition 3 (*Pareto efficiency*) A distribution E is (ex-ante) *Pareto-efficient* if there is no distribution D such that $D[u] \geq E[u]$ for all $u \in \mathcal{U}$ and $D[u] > E[u]$ for at least one $u \in \mathcal{U}$.

The notion of Pareto-efficiency expresses the impossibility of improving the satisfaction probability of some user without detriment to anyone else. Clearly any maxmin-fair distribution is Pareto-efficient, hence any solution in its support is *maximal* (with regard to set inclusion).

The notion of minmax-fairness outlined above requires that no user satisfaction can be decreased without increasing that of another user which is no worse off, or losing Pareto-efficiency.

Definition 4 A Pareto-efficient distribution F over $\mathcal{S}$ is *minmax-Pareto* (or minmax fair) for $\mathcal{U}$ if for all Pareto-efficient distributions D over $\mathcal{S}$ and all $u \in \mathcal{U}$, it holds that

$$D[u] < F[u] \implies \exists v \in \mathcal{U} \mid D[v] > F[v] \geq F[u].$$

Requiring Pareto-efficiency is redundant for maxmin-fairness, but crucial for minmax-Pareto efficiency; without it, the definition would be met by a distribution of solutions satisfying nobody (for example, a solution which always returns the empty matching).

In “Appendix A” we present analogues to Lemma 1 and Theorem 1 (Lemma 7 and Theorem 14) for minmax-fairness.

3.4 Inequality properties

The main result of this section, Theorem 2 is that, for matroid problems, the notions of minmax fairness and maxmin fairness coincide; intuitively, any excess satisfaction probability for the best-off user can be taken away from him and redistributed to others. This also implies that the maxmin-fair solution minimizes the largest gap in satisfaction probabilities; among those, it minimizes the second-largest gap, etc.

Definition 5 The *sorted inequality vector* of a distribution D over $\mathcal{S}$, written $D^{\downarrow}_{\neq}$, is the vector of all pairwise differences in the satisfaction probabilities of the elements of $\mathcal{U}$ under D, sorted in decreasing order.

Theorem 2 *For matroid problems, the following are equivalent:* (1) *D is maxmin-fair;* (2) *D is minmax-Pareto;* (3) *D is Pareto-efficient and $D^{\downarrow}_{\neq} \preceq E^{\downarrow}_{\neq}$ for all Pareto-efficient distributions E over $\mathcal{S}$.*

The proof may be found in "Appendix A".

Note that this result does not hold in general for non-matroid problems; the following shows a counterexample.

Example 6 Consider the problem instance where the set of individuals is $\mathcal{U} = \{0, 1, 2, 3\}$ and the set of feasible solutions is $\mathcal{S} = \{\{0, 1\}, \{1, 3\}, \{0, 2, 3\}\}$. Here elements 1 and 2 never appear together in a solution, so the minimum satisfaction probability cannot exceed $\frac{1}{2}$. In order to achieve $\frac{1}{2}$ we need to choose $\{0, 2, 3\}$ with probability exactly $\frac{1}{2}$; this fixes the satisfaction probabilities of 2 and 1 to $\frac{1}{2}$, and to maximize the second-smallest probability we need to pick $\{0, 1\}$ and $\{1, 3\}$ with probability $\frac{1}{4}$ each. This is the maxmin-fair distribution D_1 and its maximum inequality is $\frac{1}{4}$. However, a similar argument shows that the minmax-fair distribution D_2 is different: it uses each element of $\mathcal{S}$ with probability $\frac{1}{3}$ and has maximum inequality $\frac{1}{3}$.

Note that in this case one may verify that D_1 still minimizes maximum inequality, but by considering the complements of each element of $\mathcal{S}$, one can give a similar example where the maxmin-fair distribution does not minimize inequality.

4 A polynomial-time algorithm for maxmin-fair matching

In this section we present our main contribution: a polynomial-time algorithm for maxmin-fair matching. We present our algorithm for the *one-sided fair bipartite matching* problem. This is the special case of fair matching where:

- G is bipartite (with bipartition $V = L \dot{\cup} R$),
- the set of users is $\mathcal{U} = L$,
- there is a maximum matching covers all the right-side vertices but not all the left-side vertices (i.e., $\rho(L) = |R| < |L|$),
- there are no degree-0 vertices (which can always be removed).

This setting corresponds to the job-search setting that we use in our examples throughout the paper. We will see later (Sect. 5.2) that the general fair matching problem in non-bipartite graphs, with arbitrary user sets $\mathcal{U} \subseteq V$ and with no further restrictions, can be reduced to this special case in polynomial time. Before presenting the building blocks of our algorithm in full detail, we provide an overview of our techniques. Some results are only stated here; their proofs may be found in "Appendix B".

4.1 Overview of our techniques

We next provide an overview of how we obtain our main result: an efficient algorithm for maxmin-fair bipartite matching.

(1) The first ingredient (Sect. 4.2) is a characterization of the *fairness parameter*, i.e., the maximum satisfaction probability which can be guaranteed for every user. By using Hall's theorem we prove (Corollary 2) that the fairness parameter is determined by a "blocking" set of vertices with the smallest neighborhood-to-size ratio. Unfortunately, the proof does not lead to an efficient algorithm to find this set.
(2) Thus we proceed to write down a linear program for a fractional variant of the problem (Sect. 4.3). Inspired by a technique developed by Charikar (2000) for the *densest-subgraph problem*, we show (Lemma 3) that any fractional solution can be leveraged to find a blocking set of vertices. The neighbors of the blocking set cannot be matched to any vertex outside the blocking set in any maxmin-fair distribution. We use this fact to argue inductively (Theorem 5) the existence of a "fair decomposition" of the set of left vertices with the following property: vertices on higher levels can be allowed larger satisfaction probabilities, regardless of which edges are used to match the vertices on lower levels.
(3) Having computed the assignment probabilities x_{uv} (the probability of each pair of vertices being matched) of some maxmin-fair distribution within each block in the decomposition, we can turn each of them into an actual distribution of matchings by finding the Birkhoff–von Neumann decomposition of a doubly-stochastic matrix. Then we combine them into a single distribution.
(4) To obtain our faster algorithm (which also returns the exact optimal solution), we avoid the use of linear programming and instead present a technique to find several blocks in parallel with a single min-cut computation (Sect. 5). We show that a logarithmic number of minimum cut computations suffice to obtain the fair decomposition in full. Then we argue that given the decomposition and satisfaction probabilities, the required distribution of matchings can be found by coloring the edges of an appropriately constructed regular bipartite graph, for which task we leverage the fast algorithm of Goel et al. (2013).

4.2 Fairness parameter

We next ask the following important question: what is the minimum satisfaction probability $\pi(G)$ of a maxmin-fair distribution for G? Hall's marriage theorem gives a necessary and sufficient condition for the existence of a matching covering the whole of L, which is equivalent to having $\pi(G) = 1$.

Theorem 3 (Hall 1935) *In a bipartite graph with bipartition* (L, R)*, the set* L *is matchable if and only if* $|\Gamma(S)| \geq |S|$ *for all* $S \subseteq L$.

We show a generalization of Hall's theorem which will prove useful to characterize the fairness parameter in bipartite matching.

Theorem 4 *Let* $\{\alpha_v \mid v \in L\}$ *be reals in* $[0, 1]$. *A necessary and sufficient condition for the existence of a distribution* D *of matchings of* G *such that* $D[v] \geq \alpha_v$ *for all* $v \in L$ *is*

$$\textit{for all } S \subseteq L, \quad |\Gamma(S)| \geq \sum_{v \in S} \alpha_v. \tag{2}$$

Proof Necessity is clear because no matching can cover more than $|\Gamma(S)|$ elements of any set S, but the expected number of elements of S covered by D is $\sum_{v\in S} D[v] = \sum_{v\in S} \alpha_v$ by linearity of expectation.

For sufficiency, we may assume that all the α_v are rational because (2) is a finite set of inequalities with integral coefficients, so the maximizer of $\sum_v \beta_v$ subject to $|\Gamma(S)| \geq \sum_{v\in S} \beta_v$ and $\beta_v \geq \alpha_v$ will have $\beta_v \in \mathbb{Q}$. Let M be a suitable common denominator, so that $\alpha_u = \beta_u = n_u/M$ where $M \geq n_u \in \mathbb{N}$. Construct a graph G' with

- n_u replicas $u^{(1)}, \ldots, u^{(n_u)}$ of each $u \in L$;
- M replicas $v^{(1)}, \ldots, v^{(m)}$ of each $v \in R$;
- $V(G') = L' \cup R'$, where $L' = \{u^{(i)} \mid u \in L, i \leq n_u\}$ and $R' = \{v^{(i)} \mid v \in R, i \leq M\}$.
- $E(G') = \{(u^{(i)}, v^{(j)}) \mid (u, v) \in E(G), i \leq n_u, j \leq M\}$.

This graph is bipartite with bipartition (L', R'). Notice that vertices with $\alpha_v = 0$ have no replica in G'.

Consider (in G) the sets $A_k = \{u \in L \mid n_u \geq k\}$ for $k = 1, 2, \ldots, M$. Given k and a set $S \subseteq A_k$ let $S^{(k)} = \{u^{(k)} \mid u \in S\}$. If $A_1 = \emptyset$ the theorem is trivial. Otherwise, let H_1 denote the subgraph of G' induced by $A_1^{(1)} \cup R'$. Any subset of A_1' in H_1 is of the form $S^{(1)}$, for some $S \subseteq A_1$. Using (2) we obtain

$$|\Gamma_{H_1}(S^{(1)})| = M \cdot |\Gamma_G(S)| \geq \sum_{u\in S} n_u \geq |S^{(1)}|,$$

because $n_u \geq 1$ for $u \in A_1 \supseteq S$. By Hall's Theorem, there is a matching X_1 in H_1 covering $A_1^{(1)}$.

If $A_2 \neq \emptyset$, let H_2 denote the subgraph of $G' \setminus V(X_1)$ induced by $A_2^{(2)} \cup R'$. As we removed the edges of the matching X_1, the number of neighbours in G' of any set $S \subseteq A_2$ has decreased by at most $|S|$, so for any $S \subseteq A_2$ he have

$$|\Gamma_{H_2}(S^{(2)})| \geq M \cdot |\Gamma_G(S)| - |S| \geq \sum_{u\in S} (n_u - 1) \geq |S^{(1)}|,$$

because $n_u \geq 2$ for $u \in A_2 \supseteq S$. Hence there is a matching X_2 in H_2 covering $A_2^{(2)}$. Proceeding similarly, we obtain a set of vertex-disjoint matchings in G' such that their union is a matching X' in G' covering L'. By restricting X' to each replica of R in R', we can decompose X' into M matchings $X_1, \ldots, X_M$, each of them inducing a matching in G. Furthermore, each $u \in L$ is covered in exactly n_u of these, since X' covers L'. Thus the uniform distribution over $X_1, \ldots, X_M$ yields coverage probability $n_u/M = \alpha_u$ for each $u \in L$. □

The proof gives a maxmin-fair distribution which is uniform over a multiset of M matchings, but M may be fairly large, as large as $2^{\Omega(\sqrt{|\mathcal{U}|})}$ in some instances.

Corollary 2 *The minimum satisfaction probability in a maxmin-fair distribution for the one-sided bipartite matching problem is*

$$\pi(G) = \min\left\{\frac{|\Gamma(S)|}{|S|} \mid \emptyset \neq S \subseteq L\right\}.$$

Proof Fix a parameter $\lambda \in [0, 1]$. By Theorem 4, a distribution with satisfaction probability at least λ for all L exists if and only if $\Gamma(S) \geq \lambda|S|$ for all $S \subseteq V$. □

In "Appendix B" we prove a dual result for the maximum satisfaction probability:

Corollary 3 *The maximum satisfaction probability in a maxmin-fair distribution for the one-sided bipartite matching problem is*

$$\Pi(G) = \max\left\{\frac{|\Gamma(L)| - |\Gamma(S)|}{|L \setminus S|} \mid S \subsetneq L\right\}.$$

4.3 A compact LP formulation for the fairness parameter

Below we write a linear program for computing $\pi(G)$.

$$\begin{array}{rll} \text{minimize } \sum_{v\in R} y_v & & \\ \text{s.t. } \quad y_v - y_u \geq 0 & \forall (u, v) \in E \subseteq L \times R & \\ \sum_{u \in L} y_u = 1 & \forall u \in L & \\ y_u, y_v \geq 0 & \forall u \in L, v \in R & \end{array} \tag{3}$$

Any set $S \subseteq L$ can be represented by a feasible solution to this LP by setting $y_x = \frac{1}{|S|}$ for all $x \in S \cup \Gamma(S)$.

Lemma 2 *For any non-empty set $S \subseteq L$, there is a feasible solution to LP (3) with value $\frac{|\Gamma(S)|}{|S|}$.*

Proof Define $y_x = \frac{1}{|S|}$ for all $x \in S \cup \Gamma(S)$ and $y_x = 0$ elsewhere. Then $\sum_{u\in L} y_u = \sum_{u\in S} \frac{1}{|S|} = 1$ and for every edge $(u, v) \in L \times R$ we have either $y_u = 0$ (in which case $y_v \geq 0 = y_u$) or $y_u = 1/|S|$; the latter implies $u \in S$ and $v \in \Gamma(S)$, so $y_v = 1/|S| = y_u$. This proves feasibility. Finally, $\sum_{v\in R} y_v = \sum_{v\in\Gamma(S)} \frac{1}{|S|} = \frac{|\Gamma(S)|}{|S|}$. □

The following shows how to round an optimal solution LP (3) to obtain a set S of vertices such that $|\Gamma(S)|/|S|$ equals the optimal value. A similar technique has been used by Charikar (2000) for the densest subgraph LP.

Lemma 3 *Let $\{y_w\}_{w\in L\cup R}$ be an optimal solution to (3). Then the set $S = \{v \in L \mid y_v > 0\} \neq \emptyset$ satisfies $\frac{|\Gamma(S)|}{|S|} = \sum_{v\in r} y_v$.*

Proof Write $\lambda = \sum_{v\in R} y_v$. For any $r \in (0, 1)$, define $S(r) = \{u \in L \mid y_u \geq r\}$ and $T(r) = \{v \in R \mid y_v \geq r\}$. We show that $T(r) = |\Gamma(S(r))|$ and $|T(r)|/|S(r)| = \lambda$

for *every* $r \in (0, 1)$. To see this, observe that for any $v \in R$, $y_v \geq \max_{u \in \Gamma^{-1}(v)} y_u$. In fact in any optimal solution equality must hold: $y_v = \max_{u \in \Gamma^{-1}(v)} y_u$ for all $v \in R$; otherwise we may decrease some y_v and hence the objective function without sacrificing feasibility. Consequently,

$$
\begin{aligned}
v \in T(r) \Leftrightarrow y_v \geq r \Leftrightarrow \max_{u \in \Gamma^{-1}(v)} y_u \geq r \Leftrightarrow \\
\Leftrightarrow \exists u \in \Gamma^{-1}(v) \text{ such that } y_u \geq r \Leftrightarrow v \in \Gamma(S(r)).
\end{aligned}
$$

Recall from Lemma 2 that we can construct a solution to LP (3) from any non-empty set. Since λ is the optimal value of LP (3), for any r for which $S(r) \neq \emptyset$ we have $|T(r)|/|S(r)| \geq \lambda$, i.e., $0 \leq |T(r)| - \lambda|S(r)|$. The latter also holds if $S(r) = \emptyset$. On the other hand, if we pick r uniformly at random from $(0, 1)$, we have

$$
\begin{aligned}
\mathbb{E}_r[|S(r)|] &= \sum_u \Pr_r[u \in S(r)] = \sum_u \Pr_r[r \leq y_u] = \sum_u y_u = 1, \\
\mathbb{E}_r[|T(r)|] &= \sum_v \Pr_r[v \in T(r)] = \sum_v \Pr_r[r \leq y_v] = \sum_v y_v = \lambda,
\end{aligned}
$$

so $0 \leq \mathbb{E}_r[|T(r)| - \lambda|S(r)|] = \mathbb{E}_r[|T(r)|] - \lambda \cdot \mathbb{E}_r[|S(r)|] = \lambda - \lambda \cdot 1 = 0$, which implies that $T(r) - \lambda \cdot S(r) = 0$ almost surely when r is uniform in $(0, 1)$. Observe that $T(r)/S(r)$ is piecewise-constant in its domain (all distinct possibilities are given by taking $t = y_w$ for some $w \in L \cup R$). Moreover, for any $r \in (0, 1)$ there is some interval I of non-zero length such that for all $r' \in I$, then $S(r) = S(r')$ and $T(r) = T(r')$. Thus, any event that is a measurable function of $S(r)$ and $T(r)$ and holds with probability 1 when $r \sim U(0, 1)$ must actually hold for every $r \in (0, 1)$ as well.

Thus, $|T(r)| = \lambda|S(r)|$ for all $r \in (0, 1)$. In particular if we pick $r_0 = \min_{u \in L} y_v$, then $S(r_0) = \{v \in L \mid y_v > 0\}$ satisfies $\sum_{v \in S(r_0)} y_v = 1$, hence is non-empty, and by the above we have $|\Gamma(S(r_0))| - \lambda \cdot |S(r_0)| = 0$, as desired. □

In combination with Corollary 2, these two lemmas yield an effective method of computing $\pi(G)$:

Corollary 4 *In the one-sided fair bipartite matching problem, the fairness parameter $\pi(G)$ is equal to the optimum value of the LP in (3).*

4.4 Fair decompositions

The next ingredient towards an efficient algorithm is to find a decomposition of L according to different levels of satisfaction probability in the maxmin-fair distribution. In Fig. 2, the set of left vertices with smallest neighbor-to-size ratio is the set $B_1 = \{a_5, a_4\}$, with $\Gamma(B_1) = \{b_3\}$. By Corollary 2, the fairness parameter of the graph in the picture is $\frac{1}{2}$. But in order to actually match a_5 and a_4 with probability $\frac{1}{2}$, b_3 must be matched to one of the two every single time. Hence the edge (a_3, b_3) can never be used to in a maxmin–maxmin-fair solution. After removing B_1 and $\Gamma(B_1)$ from

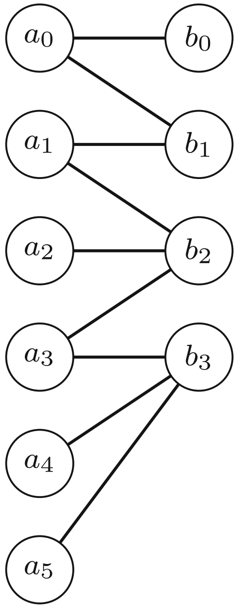

Fig. 2 A bipartite graph with blocks $B_1 = \{a_5, a_4\}$, $B_2 = \{a_3, a_2, a_1\}$ and $B_3 = \{a_0\}$ and fairly isolated sets $S_1 = B_1$, $S_2 = B_1 \cup B_2$ and $S_3 = B_1 \cup B_2 \cup B_3$

the graph, the next set of left vertices with smallest neighbor-to-size ratio is the set $B_2 = \{a_1, a_2, a_3\}$ and again we find that edge (a_0, b_1) cannot be used. The last set we find in this way is $B_3 = \{a_0\}$.

We refer to B_1, B_2, B_3 as the *blocks* of the fair decomposition; and to the increasing sequence of sets $S_1 = B_1$, $S_2 = B_1 \cup B_2$ and $S_3 = B_1 \cup B_2 \cup B_3$ as the *fairly isolated sets*. This motivates the following definitions.

For $A \subseteq L$, denote by $G|_A$ the subgraph of G induced by $A \cup \Gamma(A)$, and by G/A the subgraph of G induced by $(L \cup R)\backslash(A \cup \Gamma(A))$. Intuitively, $G|_A$ represents the subproblem where only the elements of A are important, and G/A represents the subproblem of $G|_{\overline{A}}$ where the use of neighbours of A is disallowed. For any subgraph H of G, let $\pi(H)$ (resp., $\Pi(H)$) be the minimum (resp., maximum) satisfaction probability of an element of $V(H) \cap L$ in a maxmin-fair distribution. The nonempty set $X \subseteq L$ is *fairly isolated* if $\Pi(G|_X) < \pi(G/X)$ or $X = L$. This means that every $u \notin X$ has guaranteed satisfaction larger than the largest maxmin-fair satisfaction inside X, even if we remove all possibly conflicting edges from X to $\Gamma(X)$.

Finding fairly isolated sets enables a "divide and conquer" strategy to find maxmin-fair distributions, since it turns out that matchings used inside X have no bearing on the satisfactions needed for users in $L\backslash X$. For example, if we can determine that the set $B_1 \cup B_2$ is fairly isolated, then we can work independently on $B_1 \cup B_2$ and B_3 and combine the distributions found.

With this in mind, we are ready to state our fair decomposition theorem, proved in "Appendix B":

Theorem 5 *The fairly isolated sets form a chain $S_1 \subseteq S_2 \ldots \subseteq S_{k-1} \subseteq S_k = L$. Define $S_0 = \emptyset$ for convenience and let $B_i = S_i \backslash S_{i-1}$ for $i > 0$. The following hold for all $i = 1, \ldots, k$:*

(a) *B_i is the maximal set $X \subseteq L \setminus S_i$ minimizing* $\frac{|\Gamma(X \cup S_i)| - |\Gamma(S_i)|}{|X|}$.
(b) *If $i < k$, B_i is the maximal set $X \subseteq S_{i+1}$ maximizing* $\frac{|\Gamma(S_{i+1})| - |\Gamma(S_{i+1} \setminus X)|}{|X|}$.
(c) *The satisfaction probability of every $v \in B_i$ in any maxmin-fair distribution is $\lambda_i = \frac{|\Gamma(B_i) \setminus \Gamma(S_{i-1})|}{|B_i|}$, and any $w \in \Gamma(B_i) \setminus \Gamma(S_{i-1})$ is matched to some $u \in B_i$ with probability 1.*

We call $B_1, \ldots, B_k$ the blocks *in the fair decomposition of G.*

4.5 Description of the basic algorithm

Theorem 5 and Lemma 3 suggest a line of attack to solve the one-sided fair bipartite matching problem, outlined in Algorithm 1 below. First, find the blocks $B_1, \ldots, B_k$ in a fair decomposition. Second, find a maxmin-fair distribution[1] D_i for each block B_i, using only edges that do not "cross to neighbors of lower blocks" (i.e., no edge is allowed from $u \in B_i$ to $v \in \Gamma(B_j)$ where $j < i$). Finally, combine the distributions into a single maxmin-fair distribution D, and draw a matching from it. Both our algorithms follow this general outline; they differ on how to perform steps 1 and 2. (We will discuss later (Sect. 7) an alternative implementation of step 3 which leads to distributions over a smaller number of matchings.)

Next we give the details of our first algorithm (Algorithm 2).

Algorithm 1: Outline of our polynomial-time algorithms for maxmin-fair matching

Input: Bipartite graph $G = (V, E)$ with bipartition $V = L \dot\cup R$ and with $|\rho(L)| = R$
Output: A maximum matching in G drawn from a maxmin-fair distribution

Function `MaxminFairMatching`(G, L, R)
  `/* Step 1: find a fair decomposition */`
  $B_1, \ldots, B_k =$ `FairDecomposition`(G, L, R)
  `/* Step 2: obtain a fair decomposition for each block */`
  for $i = 1, \ldots, k$ **do**
    $R_i = \Gamma(B_i) \setminus \bigcup_{j<i} \Gamma(B_j)$
    $G_i =$ subgraph of G induced by B_i and R_i
    $\mathcal{D}_i =$ `SingleBlockDistribution`(G_i, B_i, R_i)
  `/* Step 3: combine the distributions and pick a matching */`
  for $i = 1, \ldots, k$ **do**
    $M_i =$ draw a matching from $\mathcal{D}_i$
  return $\bigcup_{i=1}^{k} M_i$

Step 1: Finding a fair decomposition We will find the blocks in a bottom-up manner. To find the first block, observe the following:

[1] Each of these distributions can be represented by a list of pairs $(probability, matching)$, with the probabilities being non-negative and summing up to 1. For our second algorithm a simpler representation is possible: the distribution of matchings D_i within each block (but not for the whole graph) is uniform over some small multiset of matchings.

Lemma 4 *The maximal set minimizing $|\Gamma(X)|/|X|$ is the union of all non-empty sets X minimizing $|\Gamma(X)|/|X|$.*

Proof It suffices to show that if X, Y are non-empty sets minimizing $|\Gamma(X)|/|X|$, then $X \cup Y$ also minimizes $|\Gamma(X)|/|X|$. Indeed, suppose $\frac{|\Gamma(Y)|}{|Y|} = \frac{|\Gamma(X)|}{|X|} \triangleq \lambda$. By the submodularity of the cardinality of the neighborhood function of a graph,

$$|\Gamma(X \cup Y)| + |\Gamma(X \cap Y)| \leq |\Gamma(X)| + |\Gamma(Y)| = \lambda(|X| + |Y|).$$

Notice that $|\Gamma(X \cup Y)| \geq \lambda|X \cup Y|$ and $|\Gamma(X \cap Y)| \geq \lambda|X \cap Y|$ by definition. If any of these two inequalities were strict we would have the contradiction

$$|\Gamma(X \cup Y)| + |\Gamma(X \cap Y)| > \lambda(|X \cup Y| + |X \cap Y|) = \lambda(|X| + |Y|).$$

Hence the inequalities are not strict, and $|\Gamma(X \cup Y)| = \lambda|X \cup Y|$. □

Along with Theorem 5, this observation suggests the following method, described in the FairDecomposition method of Algorithm 2. By solving the LP in (3) and using Lemma 3, we obtain a set X minimizing $|\Gamma(S)|/|S|$. Remove X from the graph G and repeat (if G is non-empty); let Y be the new set obtained. If $\Gamma(Y)/|Y| = \Gamma(X)/|X|$, then replace X with $X' = X \cup Y$ and repeat the process of finding a minimizer Y via LP (3); this strictly increases the size of X. Eventually we will obtain a Y satisfying $|\Gamma(Y)|/|Y| > \Gamma(X)/|X|$, at which point we know that X is the maximal set minimizing $\Gamma(S)/|S|$, i.e., the first non-trivial block B_1 is X. Now remove B_1 and $\Gamma(B_1)$ from G and repeat (if applicable) to obtain $B_2, \ldots, B_k$.

Step 2: Obtaining a fair distribution for each block The idea of this step is first to calculate the assignment probabilities x_{uv} for all $u \in L, v \in R$, i.e., the probability that u is matched to v in some fixed maxmin-fair distribution F. As of yet these probabilities are unknown (and, unlike satisfaction probabilities, they need not be the same for all maxmin-fair distributions). However, we do know some conditions that they must satisfy because we know (from Theorem 5) the satisfaction probabilities of the left vertices in F, and all the right vertices need to be matched with probability 1 under our assumption that $\rho(L) = |R|$. These conditions may be expressed as linear constraints in x_{uv}, so we will find suitable values for x_{uv} via a linear program. Finally we can turn these values into an actual distribution of matchings via the Birkhoff–von Neumann decomposition. Details follow.

Consider the graph $H_i = G/\bigcup_{j \leq i} B_j$ obtained by removing all lower blocks and their neighbors. To simplify notation, rename $L \cap V(H_i)$ and $R \cap V(H_i)$ to L and R. We have $|R| \leq |L|$ and $\lambda = |R|/|L| \leq 1$. First we calculate the (as of yet unknown) probabilities x_{ij} ($i \in L, j \in R$) that each edge (i, j) is saturated (i.e., i is matched to j) in some fixed maxmin-fair distribution. Clearly $\sum_j x_{ij} = \lambda$ for each i and $\sum_i x_{ij} = 1$ for each j. Let us add a set Z of $|L| - |R|$ fictitious vertices to R and extend the domain of definition of x_{ij} so as to satisfy $x_{ij} = 1/|L|$ for each $i \in L, j \in Z$. We obtain a bipartite graph G' with $|L|$ vertices on each side; let Γ' denote its neighborhood function. Then $\sum_{v \in \Gamma'(u)} x_{uv} = 1 \; \forall u \in L, \; \sum_{u \in \Gamma'(v)} x_{uv} = 1 \; \forall v \in R \cup Z,$ and $x_{uv} \geq 0 \; \forall u \in L, v \in R \cup Z$. We can find a solution x_{uv} to these inequalities by solving a linear program.

Algorithm 2: First polynomial-time algorithm for maxmin-fair matching

Input: Bipartite graph $G = (V, E)$ with bipartition $V = L \dot{\cup} R$

Function `SmallestRatioSet(`G, L, R`)`
  Solve LP (3) for the subgraph of G induced by L and R
  $S = \{v \in L \mid y_v > 0\}$
  $\lambda = \sum_{v \in R} y_v$
  return S, λ

Function `FairDecomposition(`G, L, R`)`
  $k = 0$
  $L', R' = L, R$
  while $L' \neq \emptyset$ **do**
    $X, \lambda' =$ `SmallestRatioSet(`G, L', R'`)`
    $L' = L' \setminus X$
    if $k = 0$ or $\lambda' \neq \lambda$ **then**
      `/* Create new block, possibly incomplete */`
      $k = k + 1$
      $B_k, \lambda = X, \lambda'$
      $R' = R' \setminus \Gamma_G(B_k)$ `/* Remove neighbors of the previous block */`
    else
      `/* Merge with an existing block */`
      $B_k = B_k \cup X$
  return $B_1, \ldots, B_k$

Function `SingleBlockDistribution(`G, L, R`)`
  $F =$ a set of $|L| - |R|$ new right vertices
  $N = \{(i, j) \mid i \in L, j \in F\}$
  Add the new vertices F and new edges N to G to form G'
  `/* LP to find assignment probabilities */`
  Find non-negative values x_{uv} such that $\sum_{j \in \Gamma_{G'}(i)} x_{ij} = 1$ for all $j \in L \cup R$.
  `/* Birkhoff-von Neumann decomposition */`
  Find a distribution D of matchings using edge (u, v) with probability x_{uv}.
  Remove from each matching in D the incident to F
  return D

By the following consequence of Birkhoff–von Neumann theorem on doubly stochastic matrices (Birkhoff 1946) the quantities x_{uv} thus obtained represent the edge saturation probabilities of an actual distribution of matchings in G':

Lemma 5 *Let $\{x_{uv}\}_{(u,v)\in E}$ be non-negative numbers s.t. $\sum_{v \in R} x_{uv} \leq 1\ \forall u \in L$ and $\sum_{u \in L} x_{uv} \leq 1\ \forall v \in R$. Then a distribution over $|E| + 1$ matchings such that $\Pr_{M \in \mathcal{M}}[(u, v) \in M] = x_{uv}$ exists and may be found in polynomial time.*

We thus obtain a distribution D of matchings in G' in which each edge (u, v) is used with probability x_{uv}. If we pick each matching with its probability in D and remove from it the edges incident to the "fictitious" elements in Z, we obtain a distribution of matchings where each element i of L is matched with probability $1 - \sum_{j \in Z} x_{ij} = 1 - (|Z|/|L|) = 1 - (|L| - |R|)/|L| = \lambda$, as desired.

Step 3: Combining the distributions The last step requires combining the distributions $D_1, \ldots, D_k$, each defined for a block B_i, into a single maxmin-fair distribution for G. The simplest way is to draw $(M_1, \ldots, M_k)$ from the product distribution $D_1 \times$

$D_2 \ldots \times D_k$ and return $M_1 \cup M_2 \ldots \cup M_k$. (This is an easily samplable maxmin-fair distribution with potentially large support.)

Putting all together, we obtain the following.

Theorem 6 *Algorithm 2 is a polynomial-time algorithm for the one-sided maxmin-fair matching problem.*

5 A more efficient algorithm

The algorithm from Sect. 4 requires solving polynomially many LP subproblems. It was presented to showcase the main steps required, to introduce the fair decompositions, and to establish the existence of a polynomial-time algorithm. In this section we analyze a more efficient algorithm. It also follows each of the three steps outlined in Algorithm 1, but differs from Algorithm 2 in two key respects:

- For step 1, it finds fairly separated sets in arbitrary order, rather than bottom-up. These sets can be found by maximum flow computations in a certain graph, and a single flow computation can be used to find many new blocks in the decomposition simultaneously.
- For step 2, it uses a fast edge-coloring algorithm on a carefully constructed regular bipartite graph, allowing us to bypass the (comparatively slow) Birkhoff–von Neumann decomposition [for which the best known algorithm from Goel et al. (2013)] runs in $\omega(|V||E|)$ time).

We present pseudocode for the improved algorithm (Algorithm 3) at the end of this section. We establish the following:

Theorem 7 *Algorithm 3 solves the maxmin-fair one-sided bipartite matching problem in* $O((|V|^2 + |E||V|^{2/3}) \cdot (\log |V|)^2)$ *expected time.*

5.1 Improved step 1: finding a fair decomposition

Suppose we wish to separate L into vertices with satisfaction probability $< \lambda$ and vertices with satisfaction probability $\geq \lambda$, for some parameter $\lambda \in (0, 1)$. To this end, construct the graph $G(\lambda)$ by adding to G a source vertex s connected to every $u \in L$ with an edge of capacity λ, and a sink vertex t connected to every $v \in R$ with an edge of capacity 1; all other edges have infinite capacity.

Lemma 6 *Let κ be the value of a minimum $s - t$ cut in $G(\lambda)$. Then exactly one of the following cases holds: (a) $\kappa = \lambda|L|$ and $\pi(G) \geq \lambda$; or (b) $\kappa < \lambda|L|$ and there is a fairly-isolated subset $X \subsetneq L$ such that $\Pi(G|_X) < \lambda$. We can determine which case occurs, and obtain X in case (b), with a min-cut computation on $G(\lambda)$.*

Recall that $\pi(G)$ (resp., $\Pi(G)$) represents the minimum (resp., maximum) satisfaction probabilities in a maxmin-fair distribution for G. In either of the two cases contemplated by Lemma 6 we have "made progress" by solving a min-cut problem on $G(\lambda)$; either (a) we showed that achieving minimum satisfaction probability λ is possible, or (b) found a fair separation (and a reason why it is not possible).

Proof Consider a minimum-value $s - t$ cut in $G(\lambda)$. Because the capacities of the edges from s are no larger than any other capacity, there is always a cut C with no larger value containing no edges from L to R. C only contains edges from s to some subset $\overline{A}_L \subseteq L$ and from some subset $A_R \in R$ to t; its value is $\lambda|\overline{A}_L| + |A_R|$.

Let $A_L = L \backslash \overline{A}_L$ and $\overline{A}_R = R \backslash A_R$. Because C is an s, t-cut, there are no edges in G (or in $G(\lambda)$) between A_L and $\overline{A}_R$, so $\Gamma(A_L) \subseteq A_R$. As C is a *minimum* cut, we must in fact have $\Gamma(A_L) = A_R$ (or else cutting some edges from A_R to t is unnecessary). The value of C is $\lambda|\overline{A}_L|+|A_R|$. Furthermore, for any $X \subseteq \overline{A}_L$ we must have $|\Gamma(X)\backslash\Gamma(A_L)| \geq \lambda|X|$, for otherwise there would be a cut of smaller value

$$\lambda|\overline{A}_L \backslash X| + |\Gamma(A_L \cup X)| = (\lambda|\overline{A}_L| + |A_R|) - \lambda|X| + |\Gamma(X)\backslash A_R|.$$

So the fairness parameter $\pi(G/A_L)$ is at least λ. If $A_L = \emptyset$, this is $\pi(G)$ and we are in case (a) of the Lemma.

If $A_L \neq \emptyset$, let $C = C(\lambda)$ be the minimum cut with minimum $|A_L|$. Then A_L is unique and may be determined in linear time by picking the vertices reachable from s in the residual network of a maximum (pre)flow (Picard and Queyranne 1980). For any $Y \subseteq A_L$, we must have $|\Gamma(A_L)| - |\Gamma(A_L\backslash Y)| < \lambda|Y|$, otherwise another cut C' of at most the same same value but with $|A'_L| < |A_L|$ would exist. Hence $\Pi(G|_{A_L}) < \lambda$ by Corollary 3 which, along with the previously derived inequality $\pi(G/A_L) \geq \lambda$, states that A_L is a fairly separated set, and we are in case (b). ◻

The parametric flow algorithm of Gallo et al. (1989) can find the cuts $C(\lambda)$ in the proof of Lemma 6 simultaneously for all λ (in the sense of giving a cut for all possible $|L|-1$ "breakpoints" for λ). Its running time is asymptotically the same time as that of a single maximum-flow computation via the push-relabel algorithm of Goldberg and Tarjan (1988). However, this technique does not extend to all max-flow algorithms, and Goldberg and Tarjan (1988) is suboptimal for the graphs $G(\lambda)$. A better idea is the following (see Algorithm 3).

Start with $\lambda = 1$ and keep halving λ as long as case (a) holds in Lemma 6. The first time that (b) occurs we have found a fairly separated set X. At this point we can find recursively the blocks in the fair decompositions of $G|_X$ and G/X. The crucial insight is that *we can find both in a single recursive call*: $G|_X$ and G/X are disjoint, so min-cuts for $(G|_X)(\lambda_1)$ and $(G/X)(\lambda_2)$ are easily obtained from min-cuts for a single graph $G(\lambda_1, \lambda_2; X, \overline{X})$ containing a disjoint copy of each (except that we still keep a single source s and a single sink t).

An iterative implementation of this idea maintainins the following invariant:

(a) we keep a partition of L into $t \leq k$ subsets $T_1, \ldots, T_t$;
(b) each T_i is the union of consecutive blocks in the decomposition (in other words, it is the difference between two fairly isolated sets);
(c) we have computed lower and upper bounds λ_i and μ_i for the maxmin-fair probabilities of vertices in T_i, i.e., $[\pi(T_i), \Pi(T_i)] \subseteq [\lambda_i, \mu_i)$;
(d) these bounds satisfy $\mu_i - \lambda_i = 2^{-j}$ at iteration $j \geq 0$.

Initially, $t = 1$, $T_1 = L$, $\lambda_1 = 0$, $\mu_1 = 1$ (valid by the assumption $\rho(L) < |L|$), and $j = 0$. Construct the graph $G(\lambda'_1, \ldots, \lambda'_t; T_1, \ldots, T_t)$ where the edge capacities from s to each $u \in T_i$ are $\lambda'_i = (\lambda_i + \mu_i)/2$, and edges from $u \in T_i$ to $v \in \Gamma(T_j)$ where $j \neq i$ are deleted. With a min-cut computation in $G(\lambda'_1, \ldots, \lambda'_t; T_1, \ldots, T_t)$ we reduce the range of parameter bounds within T_i by half for each i, and possibly split T_i into two (increasing t) if we found a new fairly separated set. After the min-cut computation, obtaining the new partition of L, the new upper bounds, and removing the edges from lower blocks to higher ones takes linear time.

After $O(\log |L|)$ iterations (each performing a min-cut and a linear-time update), we have $\mu_i - \lambda_i < 1/|L|^2$ for all i, at which point we have determined the full decomposition (because each maxmin-fair satisfaction probability is of the form a/b where $a \leq b$, $1 \leq b \leq |L|$). The running time of the max-flow algorithm of Goldberg and Rao (1998) for bipartite networks with rational capacities with denominators bounded by a polynomial in $|V|$ is $O(\min(|E|^{3/2}, |E||V|^{2/3}) \cdot \log |V|)$. We obtain:

Theorem 8 *The fair decomposition of a graph $G = (V, E)$ for the one-sided fair bipartite matching problem can be found in time $O(\min(|E|^{3/2}, |E||V|^{2/3}) \cdot (\log |V|)^2)$.*

5.2 Improved step 2: obtaining a fair distribution for each block

Here we describe the procedure in Algorithm 3 to find fair distributions once the fair decomposition has been computed. As before, suppose that G itself has a single block, so $\Gamma(L) = \lambda|R|$. Let $g = \gcd(|L|, |R|)$ and $l = |L|/g$, $r = |R|/g$.

Let $G(\lambda)$ be as in Lemma 6. By the max-flow/min-cut theorem, there is a flow in $G(\lambda)$ of value $\lambda|L| = |R|$. Since the incoming edges to any $u \in L$ from s have capacity λ, the flow from s to v must be precisely λ. Let x_{uv} be the flow between $u \in L$ and $v \in R$. Then $\sum_{u \in L} x_{uv} = \lambda = \frac{l}{r}$ and $\sum_{v \in R} x_{uv} = 1$, so we found the edge saturation probabilities $\{x_{uv}\}$ of a maxmin-fair distribution.

Consider now the subgraph G' of G containing only those edges for which $x_{uv} > 0$. By Lemma 5, the same edge probabilities x_{uv} warrant the existence of a distribution of matchings in G' with satisfaction probability λ.

By the integral flow theorem (Lawler 1976), each x_{uv} may be assumed to be a multiple of $1/r$, because all capacities in G' are multiples of $1/r$; in fact any standard maximum-flow algorithm returns such a solution. Now consider the (r, l)-biregular multigraph P obtained by putting $n_{uv} = x_{uv} \cdot r$ parallel edges between $u \in L$ and $v \in R$. As in step 2 of Sect. 4.5, we add to the right side of P a set Z of $|L \backslash R|$ vertices. Joining the ith vertex of L with the jth vertex of Z whenever $i \equiv j \bmod g$, we obtain from P a graph P' which is bipartite and l-regular.

Any bipartite graph of maximum degree l is l-edge-colorable so that no two adjacent edges share a color by Kőnig's theorem [see Lovász and Plummer (2009)]. Each color class is a matching, so there are l matchings in P' covering each $u \in L$ exactly l times in total. Cole et al. (2001) give an algorithm to color regular bipartite graphs in time $O(m \log r) = O(m \log |\Gamma(L)|)$, where m is the number of edges of P'; in our case $m = O(l \cdot |E(G)|)$. Goel et al. (2013) give a randomized algorithm to color l-regular bipartite graphs in expected time $O(ln^2 \log^2(n))$, where n is the number of vertices of P'; in our case $n = O(|L|)$ and we can use the crude bound $l \leq |L|$, so it runs

Algorithm 3: Improved (faster) polynomial-time algorithm for maxmin-fair matching

Input: Bipartite graph $G = (V, E)$ with bipartition $V = L \dot{\cup} R$

Function `FairDecomposition`(G, L, R)
- $t = 1$ `/* number of sets */`
- $T_1 = L$
- $\lambda_1, \mu_1 = 0, 1$
- **while** $\mu_1 - \lambda_1 \geq \frac{1}{|L|^2}$ **do**
  - Construct the graph $G' = G\left(\frac{\lambda_1+\mu_1}{2}, \dots, \frac{\lambda_t+\mu_t}{2}; T_1, \dots, T_t\right)$ as in the discussion preceding Theorem 8
  - Run a max flow algorithm on G'
  - X = set of vertices reachable from s in the residual flow
  - $p = t$
  - **for** $i = 1, \dots, p$ **do**
    - $X_i = X \cap T_i$
    - **if** $X_i \neq \emptyset$ **then**
      - `/* Separation found; split` T_i `into two */`
      - $T_{t+1}, \lambda_{t+1}, \mu_{t+1} = T_i \setminus X_i, \frac{\lambda_i+\mu_i}{2}, \mu_i$
      - $T_i, \lambda_i, \mu_i = X_i, \lambda_i, \frac{\lambda_i+\mu_i}{2}$
      - Remove from G the edges between T_{t+1} and $\Gamma(T_i)$
      - $t = t + 1$
    - **else**
      - `/* Separation not found; update lower bound on` $\pi(T_i)$ `*/`
      - $\lambda_i = \frac{\lambda_i+\mu_i}{2}$
- **return** $T_1, \dots, T_t$

Function `SingleBlockDistribution`(G, L, R)
- $g = \gcd(|L|, |R|)$
- $l = |L|/g$
- $r = |R|/g$
- Construct the graph $H = G(\lambda)$ as in Lemma 6
- Find a maximum flow in H; let x_{uv} denote the flow between $x \in L$ and $v \in R$
- Construct a multigraph P with $x_{uv} \cdot r$ edges between each pair $(u, v) \in L \times R$
- F = a set of $|L| - |R|$ new right vertices
- Add to the right side of P the vertices in F
- $N = \{(i, j) \mid i \in L, j \in F, i \equiv j \pmod{r}\}$
- Add the edges in N to P
- $C_1, \dots, C_l$ = color classes in an l-coloring of the edges of P
- Remove from $C_1, \dots, C_l$ the edges incident to F
- D = the uniform distribution over $C_1, \dots, C_l$
- **return** D

in $O(|L|^2 \log^2(|L|))$. If we remove the "fictitious" vertices in Z from each of these matchings, we are left with a multiset of l matchings in G covering each $u \in L$ exactly r times. The uniform distribution over them is thus maxmin-fair for G.

Now consider the case that the decomposition of G has several blocks $B_1, \dots, B_k$. The values x^i_{uv} for all blocks i can be computed from a single maximum-flow computation in $G(\lambda_1, \dots, \lambda_k; B_1, \dots, B_k)$ if we know the blocks and each satisfaction probability λ_i. Then each corresponding coloring can be found in time $O(n_i^2 \log^2(n_i))$; summing these running times and noticing that $\sum_i n_i^2 \leq |L|^2$, we deduce:

Theorem 9 *Given the fair decomposition, a maxmin-fair distribution for all blocks in it can be found in* $O(|V|^2 \cdot (\log |V|)^2)$ *expected time and* $O(|V||E| \log |V|)$ *deterministic time after a max-flow computation.*

Putting all together yields Theorem 7.

6 Generalization to non-bipartite graphs

Recall that so far we have concerned ourselves with the one-sided fair bipartite matching problem, i.e., the special case of fair matching where G is bipartite (with bipartition $V = L \dot{\cup} R$) and the set of users is $\mathcal{U} = L$.

Notably, this special case can encode any other matching problem, and moreover we can make the simplifying assumption that L is matchable and larger than R. To show this, we make use of the following result from transversal theory.

Theorem 10 (Edmonds and Fulkerson 1965) *For any graph G there exists a bipartite graph H with bipartition (L_H, R_H) such that $L_H = V(G)$ and the collection of matchable subsets of $V(G)$ in G equals the collection of matchable subsets of L_H in H.*

This is normally stated as “any matching matroid is transversal”. The construction of H in Theorem 10 can be carried out in polynomial time [see Triesch (1992) for a simple proof]. Hence the case of non-bipartite G can be reduced to the one-sided bipartite case. A similar remark applies to general user sets $\mathcal{U} \subseteq V$, as we can remove from L_H the elements of $V \backslash \mathcal{U}$, which has no effect on the collection of matchable subsets of $\mathcal{U}$ in H.

We make the additional simplifying assumption that R is matchable. If not, find an arbitrary maximum matching of G and remove from R all unmatched vertices. Let R' denote the remaining vertices.

Theorem 11 (Dulmage and Mendelsohn 1958) *If both $A \subseteq L$ and $B \subseteq R$ are matchable in a bipartite graph G with bipartition (L, R), then $A \cup B$ is also matchable in G.*

It follows that for each distribution of matchings of G there is another distribution with the same coverage (satisfaction) probabilities for L and covering only elements of $L \cup R'$. Note that the coverage probability of each $v \in R'$ in this distribution is 1. The case where $\rho(L) = |R|$ is easily handled separately (any maximum matching algorithm is maxmin-fair in this case), yielding the following (see “Appendix B” for details):

Theorem 12 *The fair matching problem on arbitrary graphs with arbitrary user sets $\mathcal{U} \subseteq V$ can be reduced in polynomial time to the one-sided fair bipartite matching problem on graphs where $\rho(L) = |R| < |L|$.*

Proof Let $\mathcal{A}$ be a maxmin-fair algorithm for one-sided bipartite matching problem described. Given a graph G and a user set $\mathcal{U}$, we

(1) construct H as in Theorem 10;
(2) remove from L_H the elements of $V(G) \setminus \mathcal{U}$;
(3) find an arbitrary maximum matching M and remove from R_H the elements not covered by M, using any polynomial-time maximum matching algorithm.
(4) find a fair one-sided bipartite matching by either (a) using $\mathcal{A}$ on the resulting graph if $|M| < |L_H|$ or (b) returning M if $|M| = |L_H|$;
(5) given the solution S found at the previous step, return a matching in G covering the same vertices as S using an explicit algorithm for Berge's theorem (Edmonds 1965).

[Step (5) is technically redundant as we identify solutions with sets of matchable users, but is included for clarity.]

It is plain to see that the resulting distribution is maxmin-fair for the general problem if and only if $\mathcal{A}$ is maxmin-fair for the one-sided problem. All steps run in polynomial time, possibly excluding the call to $\mathcal{A}$ itself. $\square$

Combining Theorems 7 and 12, we obtain the following.

Theorem 13 *The maxmin-fair matching problem on general unweighted graphs is solvable in polynomial time.*

7 On transparency and practical deployment

Even a provably fair algorithm might still be perceived by the average user as a black-box outputting an arbitrary solution. For the sake of transparency and accountability, it can be interesting to publish all the solutions in a maxmin-fair distribution (along with their respective probabilities). Once a complete fair distribution is published, convincing any user u of fair treatment amounts to:

(1) letting u verify independently the fairness guarantees of the distribution (for this it is also possible to output a short certificate, based on the fair decomposition, of the fact that no higher probability for u is possible in a maxmin-fair distribution); and
(2) picking one of the published solutions at random, via any fair and transparent lottery mechanism or coin-tossing protocol (this is the only stage where randomness plays a role, as the distribution of matchings itself can be found deterministically).

One difficulty is the potentially large support size of the maxmin-fair distribution, which could prevent publication. An interesting question is if we can produce a maxmin-fair distribution with small support. It turns out that for matchings, $|L| - 1$ solutions always suffice; although the actual number can be substantially smaller in practice (as shown in Sect. 8).

Let us discuss how to modify our algorithm so as to find a maxmin-fair distribution F using at most $|L| + 1 - k$ matchings, where k is the number of non-trivial blocks in Theorem 5. (This could replace step 3 of Algorithm 1.) When $k = 1$, the technique from step 2 of Algorithm 3 gives a multiset of $l \leq |L|$ matchings.

Consider the case $k = 2$, which implies our claim for larger k by induction. Suppose D (resp., D') chooses matching $M_i, i \in [r]$ on B_1 (resp., $N_j, j \in [t]$ on B_2) with probability p_i (resp., q_j). (Here $B_1 \cap B_2 = \emptyset$.) A simple greedy algorithm can construct a distribution Z of matchings in $B_1 \cup B_2$ such that $D[u] = Z[u]$ for $u \in B_1$ and $D'[u] = Z[u]$ for $u \in B_2$ with at most $r + t - 1$ matchings, as follows.

Keep indices $i \in [r]$, $j \in [t]$ and let S denote a set of (probability, matching) pairs, which will define the desired distribution at the end. At the outset $S = \emptyset$ and $i = j = 1$; at each iteration we add to S the new pair $(\delta, M_i \cup N_j)$ where $\delta = \min(p_i, q_j)$. We decrement p_i and q_j by δ and increment i (resp., j) if p_i (resp, q_j) vanishes. The process terminates when i and j reach the end of their range, at which point $|S| = r + t - 1$ and all probabilities in S sum up to 1.

We note, however, that this procedure may produce some matchings with very small probabilities, so the precision needed to specify a maxmin-fair distribution exactly will grow.

8 Experimental evaluation

We evaluate the practical performance of our fair matching algorithm by measuring its running time and its ability to scale to large graphs, and analyzing the distribution of maxmin-fair satisfaction probabilities and how they compare with those from two baselines. We also describe the features of the fair decompositions obtained.

Reproducibility Our code is available at https://github.com/elhipercubo/maxmin_fair_bipartite_matching.git. It implements the improved algorithm from Sect. 5, with some implementation choices described below. It was compiled with `g++` using `-O3` optimizations and run on a dual-core Intel i7-7560U CPU (2.40 GHz) with 16Gb RAM.

Datasets We used publicly-available bipartite graphs of various types, sizes and domains: all the graphs are already bipartite at the source repository, so that no pre-processing was needed. Table 2 reports their main characteristics.

Methods tested We compare the following four methods to output maximum matchings:

- (UF) *Unfair* A standard maximum matching algorithm using maximum flows, optimized for runtime using the techniques from Cherkassky and Goldberg (2004). We use it for runtime comparisons only, because such a deterministic mechanism is inherently unfair as argued in Sect. 1.
- (MF) *Maxmin fair* Our mechanism, using the improved algorithm from Sect. 5.
- (PS) *Probabilistic serial* [From Bogomolnaia and Moulin (2001)]: the goal here is to find a set of edge flows from L to R which can be converted into a matching distribution by using the Birkhoff–von Neumann decomposition.
  PS attempts to find a fair flow via a greedy algorithm, as follows: each user $u \in L$ sends flow at the same fixed rate, sharing this rate equally among her neighbours. When the outgoing flow of $u \in L$ (or the ingoing flow of $v \in R$) reaches 1, remove u (or v). Repeat while there are edges remaining.
  Unlike MF, this mechanism is not Pareto-efficient (i.e., it does not necessarily

Table 2 Datasets used: code, number of left and right nodes ($|L|$, $|R|$), number of edges ($|E|$), maximum matching size (ρ). Available at: http://konect.uni-koblenz.de/networks/

Dataset (code)	$\|L\|$	$\|R\|$	$\|E\|$	ρ
actor-movie (AM)	127,823	383,640	1,470,404	114,762
pics-ti (Vui)	82,035	495,402	2,298,816	67,608
citeulike-ti (Cti)	153,277	731,769	2,411,819	120,125
bibsonomy-2ti (Bti)	204,673	767,477	2,555,080	152,757
wiki-en-cat (WC)	1,853,493	182,947	3,795,796	179,546
movielens (M3)	69,878	10,677	10,000,054	10,544
flickr (FG)	395,979	103,631	8,545,307	96,866
dblp-author (Pa)	1,425,813	4,000,150	8,649,016	1,425,803
discogs-aff (Di)	1,754,823	270,771	14,414,659	248,796
edit-dewiki (de)	425,842	3,195,148	57,323,775	355,045
livejournal (LG)	3,201,203	7,489,073	112,307,385	2,171,971
trackers (WT)	27,665,730	12,756,244	140,613,762	4,006,867
orkut (OG)	2,783,196	8,730,857	327,037,487	1,980,077

return maximum matchings), but like MF, a single run of the mechanism can be used to output all satisfaction probabilities.

(RP) *Random priority* [See Bogomolnaia and Moulin (2001)]: it finds a matchable set of vertices as follows: Let $S = \emptyset$. Process all users in random order, adding user u to S if $S \cup \{u\}$ is matchable. Return a matching covering the final set S. Like MF, this mechanism is Pareto-efficient, but unlike MF, a single run of the mechanism only outputs a single matching and hence cannot be used to compute all satisfaction probabilities.

The latter two methods arose from work in economics in a different setting: randomized assignments on full bipartite graphs with *ordinal preferences*, i.e., where every $u \in L$ has a full ranking of all possible partners $v \in R$, and the goal is to design mechanisms which are ordinally efficient. (By contrast, in our setting the graph is not complete but there are no ordinal preferences: each user considers all of its neighbours equally desirable.) However they can be naturally applied in our context as well.

Implementation We used the improved algorithm from Sect. 5. For max flow computations we chose the highest-label push-relabel algorithm of Goldberg and Tarjan (1988), which performs best with the gap heuristic from Cherkassky and Goldberg (1997). We follow the techniques from Cherkassky and Goldberg (2004): efficient gap detection is done via bucket lists of active nodes at each level (Cherkassky and Goldberg 1997), and we arrange edges from/to the same vertex consecutively to take advantage of cache locality. We avoid floating-point computations by using exact integral multipliers.

For reasons of simplicity and/or practical efficiency, our implementation departs from the pseudocode in Algorithm 3 in the points below. None of these changes affect correctness.

Table 3 Characteristics of fair decompositions: number of blocks (k), edges used (e_1), number of matchings in the fair distribution (M)

Dataset	k	e_1	M
AM	194	143,425	13,762
Vui	72	84,003	1,068
Cti	151	157,744	2,726
B	179	212,667	4,119
WC	1468	1,883,431	350,518
M3	245	92,841	52,332
FG	924	435,612	109,242
Pa	2	1,425,813	2
Di	1117	1,784,259	305,104
de	163	432,472	5,596
LG	1480	3,314,628	302,410
WT	3612	27,842,321	16,548,387
OG	2266	3,041,112	224,738

- In FairDecomposition (line 8), only a pre-flow algorithm [the first phase of Goldberg and Tarjan (1988)] is run. This always suffices to find min-cuts (not max-flows) and thus fairly separated sets, and can halve the runtime.
- In FairDecomposition (line 7), instead of setting $\lambda'_i = (\lambda_i + \mu_i)/2$ when building the graph $G(\lambda'_1, \ldots, \lambda'_t; T_1, \ldots, T_t)$, we set it to $|\Gamma(T_i)|/|T_i|$. That is, our implementation guesses optimistically that T_i is actually a single block in the decomposition, with the pre flow computation used to verify that guess or split the block in two. This also allows us to change the terminating condition (line 6) and stop earlier: rather than stopping when λ_i and μ_i are very close, which may occur long after the full decomposition has been found, we stop when no block is split. This new choice for λ'_i may invalidate our theoretical bound on the number of flow computations required, but it makes the code much faster in practice.
- In SingleBlockDistribution (lines 27–32), we do not always build P because P may be quite large for some blocks with a small number of right vertices (where $r \ll l$), due to the fictitious edges added. Rather, we first attempt to find l disjoint matchings of size r sequentially (in arbitrary order) without including fictitious vertices/edges. This often succeeds and, when it does, gives a correct coloring. When this fails, we build P and proceed to find the coloring as described.
- In SingleBlockDistribution (line 32), we do not use Goel et al's edge coloring algorithm on P. Rather, we find l matchings of size l one by one. This simplifies the implementation, but may impact performance and the runtime bound.

8.1 Fair decompositions: characteristics

Table 3 shows the number of blocks k in the fair decomposition (for informative purposes), the number of distinct edges e_1 used in a maxmin-fair distribution F, and the number of matchings M in the support of F. As we anticipated in Sect. 7 the number of matchings needed for a fair distribution (M) is in practice much smaller than $|L|$. Another observation is that e_1 exceeds $|L|$ only slightly. This is a measure of the storage needed to publish a summary containing the fair decomposition, the satisfaction probabilities, and the probability of each edge being used in the matching, which can be verified independently. (Publishing an explicit list of M matchings of size ρ explicitly would take much more space as many of these matchings share many edges.)

8.2 Running time

In Table 4 we present runtimes of all four methods for the datasets considered. Dashes indicate times above 1 h. We report user times; the real times are within 2% of these in all cases except for OG, where the memory needs for graph and data structures exceeded the RAM available (16Gb), causing excessive disk swapping.

As MF has two clearly differentiated parts, we analyze two different runtimes:

- Time to compute the satisfaction probabilities of each user, and the probability of using each edge in the maxmin-fair distribution (step 1, finding the fair decomposition), reported in column MF_1;
- Total time, including all of the above plus the time to find an edge coloring for each block and the list of matchings for each block (step 2), reported in column MF. (The time to draw a matching given this list, step 3, is negligible.)

No clear pattern emerges as to which of these two phases is faster in practice. As can be observed, in some instances (Vui, Cti, B, Di, de, LG, OG) the time is dominated by the first phase, wheres in others (AM, M3 and WT) the total time is much larger than

Table 4 Running time (in seconds) of UF, MF, PS and RP

Dataset	UF	MF_1	MF	PS_1	RP
AM	0.34	1.812	9.309	832	11.27
Vui	0.38	1.206	1.251	717	0.39
Cti	0.42	1.731	1.820	1443	0.59
B	0.47	2.152	2.293	1764	0.74
WC	1.68	17.57	23.036	–	68.17
M3	1.08	11.12	319.13	485	72.18
FG	1.16	15.17	25.80	–	688.21
Pa	2.99	5.96	6.908	–	3.20
Di	3.10	23.37	24.714	–	67.83
de	6.99	21.67	22.007	–	52.84
LG	26.05	103.59	108.311	–	–
WT	51.92	444.71	2270.81	–	–
OG	98.20	370.76	381.524	–	–

The $_1$ subscript refers to the time to compute assignment probabilities in the solution without converting them into a distribution of matchings (only meaningful for MF and PS). Dashes indicate running times above 1 h the time for phase 1 only; in the latter cases the exact requirement of maxmin-fairness forces the algorithm to need a large number of matchings for some blocks, increasing the time for step 2. It seems likely that a more relaxed requirement of approximate fairness could lead to vast improvements in the runtime of step 2.

Similarly, for PS we differentiate between the time to compute the probability of each edge being used (column PS_1), and the total time including the former plus the time to find a distribution of matchings which agrees with those probabilities. However, this last step is much slower in the case of PS because in this case we need to use the full Birkhoff–von Neumann decomposition, instead of exploiting the degree regularity conditions of the blocks to find edge colorings as we do for MF. Because existing implementations of the Birkhoff–von Neumann decomposition do not scale even for the smaller graphs tested, we decided to omit the second phase of PS (which is not required to analyze its fairness properties).

As is to be expected, the unfair algorithm UF is the fastest. As for the others, the only one which can be run to completion within 1 h in all datasets is ours (MF). Its runtime is usually a handful of seconds except for the very large graphs, where it is in the order of minutes (37 min at most, attained for the graph WT). We can see that PS is the most computationally expensive, as many iterations of its main loop are required to reach convergence, and each iteration takes linear time. Finally, the runtime of RP is generally comparable to that of MF on small and medium-size graphs, outperforming it on many of the smaller graphs, but RP becomes slower for larger graphs; despite the additional complexity of MF, the priority mechanism of RP precludes the use of the push-relabel max-flow algorithms, and also limits the number of simultaneous augmenting paths which can be found during a single graph search in augmenting-path algorithms. Notice that these are runtimes for a single run; if the satisfaction probabilities need to be computed then it becomes necessary to run RP a large number of times, slowing it down considerably. (This is not the case for MF or PS.)

8.3 Satisfaction probability comparison

Next we analyze the satisfaction probabilities produced by maxmin-fair matchings (MF) and compare with the probabilistic serial and random-priority mechanisms. Note that the exact determination of the satisfaction probabilities of RP is computationally infeasible. To approximate them, we run RP a total of $T = 1000$ times with independent uniformly random permutations. (Note, however, that this does not give a good estimate of probabilities below $1/T$.)

For this comparison we focus on the smaller graphs, *due to the limited scalability of PS (which needs a large number of iterations in its main loop, each taking linear time) and RP (which needs to be run T times to approximate the satisfaction probabilities).*

Finally, Table 5 reports the distribution of satisfaction probabilities: minimum value ($\lambda_{\min}$), quantiles, percentage of users with satisfaction 1 ($per1$), and Nash welfare (N_0), the geometric mean of utilities (satisfaction probabilities in our setting). Nash welfare is a standard measure of fairness when allocating divisible resources (Caragiannis et al. 2016). As in Brânzei et al. (2017), we also study the generalization of Nash welfare using power means (for a parameter $p \in \mathbb{R}$):

Table 5 Distribution of maxmin-fair satisfaction probabilities

Dataset	$\lambda_{\min}$	$\lambda_{25\%}$	$\lambda_{50\%}$	$\lambda_{75\%}$	$per1$	N_0
AM	0.156	1	1	1	76.18	0.860
Vui	0.0208	0.5	1	1	69.16	0.743
Cti	0.01	0.5	1	1	65.42	0.670
B	0.0128	0.5	1	1	58.63	0.630
WC	5.52e−4	0.0149	0.0417	0.1	2.17	0.0366
M3	0.0298	0.0833	0.137	0.1798	0.49	0.121
FG	0.001012	0.116	0.2	0.254	6.04	0.161
Pa	0.5	1	1	1	99.9999	0.99999
Di	0.000025	0.0135	0.0588	0.167	4.28	0.0377
de	0.00334	0.667	1	1	74	0.718
LG	1.83e−4	0.2	1	1	59.43	0.326
WT	4.38e−7	4e−6	1.26e−4	0.0294	10.89	0.000366
OG	0.00453	0.333	1	1	56.2	0.583

$$N_0(D) = \left(\prod_{u \in \mathcal{U}} D[u]\right)^{1/|\mathcal{U}|}, \tag{4}$$

$$N_p(D) = \left(\frac{\sum_{u \in \mathcal{U}} D[u]^p}{|\mathcal{U}|}\right)^{1/p}. \tag{5}$$

When $p = 1$, $N_p(D)$ is the mean satisfaction probability, which equals $\rho/|L|$ for any Pareto-efficient mechanism. Taking the limit in (5) as $p \to 0$ one obtains (4) (Hardy et al. 1952), justifying the notation N_0 for (standard) Nash welfare. Taking the limit as $p \to -\infty$ yields $\min_{u \in \mathcal{U}} D[u]$, which by definition is maximized by MF.

Table 6 shows these metrics for the three mechanisms tested, on those graphs where PS terminated in 8 h. Notice that $N_1(\mathsf{MF}) = N_1(\mathsf{RP}) > N_1(\mathsf{PS})$, confirming that MF and RP are Pareto-efficient but PS is not. The generalized welfares for $p < 1$ are computed exactly for MF and PS, but estimated from the empirical probabilities after T samples for RP. (For $p = 0$ we replace each empirical probability q by $\max(q, 1/T)$ so that the estimate is non-zero.) MF comes out on top for all (generalized) Nash welfares in all instances, in accordance with a result of Bogomolnaia and Moulin (2004). Interestingly, $N_0(\mathsf{RP})$ is typically within 1% of $N_0(\mathsf{MF})$ (as both solutions result in a large proportion of users with high satisfaction), but for smaller p the gap can widen to as much as 22% for $p = -5$, in accordance with the fact that MF was designed to provide better guarantees to low-satisfaction users.

Table 7 shows the expected fraction of satisfied users among the bottom $t\%$ using each method, for $t = 1, 5, 10$, and 20. Again, our method always gives the highest values, the gap with the second best being as high as 50% in some instances where $t = 1$.

Table 8 reports the variance of the logarithms of satisfaction probabilities over uniformly random users, as a measure of inequality. (Taking logarithms penalizes wildly varying ratios of satisfaction probabilities.) We see that MF, which minimizes

Table 6 Generalized Nash welfare of MF, PS and RP. Larger is better

Dataset	AM	Vui	Cti	B	M3
$N_1(MF)$	0.898	0.824	0.784	0.746	0.151
$N_1(PS)$	0.796	0.775	0.739	0.702	0.147
$N_1(RP)$	0.898	0.824	0.784	0.746	0.151
$N_0(MF)$	0.860	0.743	0.670	0.630	0.121
$N_0(PS)$	0.695	0.684	0.618	0.580	0.0824
$N_0(RP)$	0.855	0.739	0.667	0.622	0.117
$N_{-1}(MF)$	0.797	0.585	0.470	0.445	0.0927
$N_{-1}(PS)$	0.524	0.522	0.419	0.395	0.0452
$N_{-1}(RP)$	0.780	0.579	0.460	0.430	0.0868
$N_{-2}(MF)$	0.699	0.339	0.232	0.244	0.0720
$N_{-2}(PS)$	0.335	0.300	0.210	0.214	0.0285
$N_{-2}(RP)$	0.659	0.332	0.228	0.230	0.0650
$N_{-5}(MF)$	0.409	0.0822	0.0431	0.0573	0.0486
$N_{-5}(PS)$	0.100	0.0767	0.0429	0.025	0.0165
$N_{-5}(RP)$	0.333	0.0793	0.0429	0.0553	0.0397

Table 7 Fraction of satisfied users among the bottom $t\%$

Dataset	AM	Vui	Cti	B	M3
$MF, t = 1$	0.189	0.0610	0.0481	0.0459	0.0298
$PS, t = 1$	0.0640	0.0528	0.0417	0.039	0.00430
$RP, t = 1$	0.147	0.0518	0.0456	0.0439	0.0198
$MF, t = 5$	0.282	0.158	0.100	0.0970	0.0298
$PS, t = 5$	0.118	0.130	0.0870	0.0825	0.00605
$RP, t = 5$	0.256	0.144	0.0045	0.0914	0.0252
$MF, t = 10$	0.389	0.231	0.151	0.143	0.0338
$PS, t = 10$	0.167	0.192	0.128	0.123	0.00883
$RP, t = 10$	0.363	0.216	0.139	0.133	0.0290
$MF, t = 20$	0.513	0.344	0.238	0.221	0.0381
$PS, t = 20$	0.259	0.287	0.204	0.193	0.0157
$RP, t = 20$	0.506	0.326	0.229	0.212	0.0353

social inequality in the sense of Definition 5, also tends to minimize this quantity in all datasets tested.

9 Related work

To the best of our knowledge, we are the first to study computationally efficient randomized maxmin-fair matching algorithms, and to offer a general definition of fairness for general search problems.

Table 8 Inequality measure: variance of log-satisfaction probabilities. Smaller is better

dataset	AM	Vui	Cti	B	M3
Var[log(MF)]	0.112	0.296	0.454	0.475	0.491
Var[log(PS)]	0.385	0.349	0.518	0.534	1.391
Var[log(RP)]	0.133	0.304	0.475	0.496	0.556

The work of Bogomolnaia and Moulin (2004) on random matching under dichotomous preferences is closely related to ours: they define an *egalitarian solution* and show that it is envy-free, strategy-proof and group-strategy-proof with respect the set of right or left vertices. As the authors note, they do not provide an axiomatic characterization of their solution; rather, their definition of egalitarian is expressed in terms of a specific algorithm and is thus not easily generalizable to other search problems. By contrast, our definition of distributional maxmin-fairness applies to any search problem with non-unique solutions and, in the special case of bipartite matchings, is equivalent to the egalitarian solution. In Bogomolnaia and Moulin (2004) two simple algorithms are proposed to find egalitarian matchings, both of them running in exponential time; our work yields a practical polynomial-time algorithm for the problem. We found no efficient algorithms or practical implementations of the egalitarian mechanism prior to our work.

Building on Bogomolnaia and Moulin (2004) and Roth et al. (2005) propose an egalitarian mechanism for the exchange of donor kidneys for transplant. McElfresh and Dickerson (2018) propose a tradeoff between fairness and a utilitarian objective function in kidney exchange programs. Kamada and Kojima (2015) study randomized matching mechanisms for the design of matching markets under distributional constraints; their setup contains full bipartite graphs equipped with complete and strict preference relationships. Teo and Sethuraman (1998) prove the existence of a "median" deterministic solution to the stable matching problem which is fair to everyone, but finding a polynomial-time algorithm remains an open problem. Cheng (2010) presents a technique to approximate the median stable matching.

In the area of resource allocation problems, several works investigate the equitable distribution of divisible resources in networks. The work of Ichimori et al. (1982) considers a minmax-style optimization function, whereas Katoh et al. (1985) considers allocation problems so that the maximum of profit differences is minimized; none of these consider distributions of several solutions. Bansal and Sviridenko (2006) give approximation algorithms for the Santa Claus problem, where a number of indivisible presents are to be distributed among kids who have different values for different presents, and the goal is to maximize the minimum happiness of a kid. Bertsimas et al. (2011) introduce the price of fairness in resource allocation problems. A substantial amount of work has also been devoted to cake-cutting algorithms and their strategic and incentive properties: see Klamler (2010), Brams et al. (2006) and Edmonds and Pruhs (2006) and the references therein.

Several authors have studied lexicographically optimal flows in networks (which could be used in place of Step 1 of our algorithms): Megiddo (1977) designed an algorithm with running time $O(n^5)$, whereas Brown (1979) proposed a polynomial-time algorithm requiring n max flow computations. On the other hand, the parametric flow algorithm of Gallo et al. (1989) can be used to find fair decompositions with a single max flow, but is not compatible with the max flow algorithm of Goldberg and Rao (1998). None of these methods can be used to match the runtime of our algorithm to find fair decompositions.

The bulk of the research in the area of algorithmic bias and fairness has mainly focused on avoiding discrimination against a sensitive attribute (i.e., a protected social group) in supervised machine learning (Dwork et al. 2012; Feldman et al. 2015; Corbett-Davies et al. 2017. Most of this literature focuses on *statistical parity*, or group-level fairness, i.e., the difference in having a positive outcome for a random individual drawn from two different subpopulations (e.g., men and women). Feldman et al. (2015) propose to repair attributes so as to maintain per-attribute within-group ordering while enforcing statistical parity, so that a single decision threshold applied to the transformed attributes would result in equal success rate among the two different groups. Corbett-Davies et al. (2017) reformulate algorithmic fairness as constrained optimization in the context of criminal justice: the objective is to maximize public safety while satisfying formal fairness constraints designed to reduce disparities. Dwork et al. (2012) provide examples showing that statistical parity alone is not sufficient for fairness, and study a randomized solution for classifiers to guarantee that "similar individuals are treated similarly" in an expected sense. The idea that more qualified individuals should be chosen preferentially is present in the work of Joseph et al. (2016), who study fairness in multi-armed bandit problems. Pedreschi et al. (2008) introduced the related data mining problem of discovering discrimination practices in a given dataset containing past decisions; if such a dataset is used as training set for a machine learning model, the bias detected can be fixed before the learning phase (Kamiran and Calders 2011; Zliobaite et al. 2011). Heidari et al. (2019) show that many existing definitions of algorithmic fairness, such as predictive value parity and equality of odds can be viewed as instantiations of economic models of equality of opportunity. Heidari et al. (2018) study a welfare-based measure of fairness for risk-averse individuals, and derive an efficient mechanism for bounding individual-level inequality.

Finally, maxmin-fairness (in a non-distributional sense) as an objective is used for flow control in networks (Coluccia et al. 2012; Bertsekas et al. 1992). In the context of non-discrimination, the concept dates back at least to Rawls's theory of justice (Rawls 1971), where a "difference principle" is advocated whereby social and financial inequalities are required to be to the advantage of the worst-off. In Rawls's distributive justice, social measures should be designed so as to bring the greatest benefit to the least-advantaged members of society, in order to maximize their prospects.

10 Conclusions

In this paper we study the problem of algorithmic fairness towards the elements that may or not be included in a solution of a matching problem. This is particularly (but not exclusively) important when these elements are humans. Towards this goal, we propose the *distributional maxmin fairness* for randomized algorithms. A series of theoretical results characterize maxmin-fair distributions and pave the road to our practical contribution: an exact polynomial-time algorithm for maxmin-fair bipartite matching, which scales to graphs with millions of vertices and hundreds of millions of edges. We also discussed methods for the transparent and accountable real-world deployment of our framework.

Regarding future work, it would be interesting to consider notions of approximate fairness intended to deal with optimization problems, where solutions may have different business value, possibly unrelated to satisfaction probabilities. The goal could be to reach a compromise between fairness and expected business value. It would be desirable to be able to find approximately maxmin-fair distributions more quickly than exact maxmin-fair distributions; we leave this as an open problem. Another interesting question is whether our methods can be extended to handle online matching/streaming settings and/or graphs which do not fit into main memory. Finally, future work may consider other notions of fairness for randomized algorithms for search, ranking and other learning problems.

Appendix A: Proofs for Sect. 3: fairness and social inequality

Preliminaries In order to prove Theorem 2, we need to recall some additonal facts about matroids [refer to Lawler (1976) for details]. The rank function $\rho\colon 2^L \to \mathbb{N}$ of a matroid is monotone submodular, meaning that for all $S, T \subseteq L$, it holds that $0 \le \rho(S \cup T) - \rho(S) \le \rho(T) - \rho(S \cap T)$. The *dual matroid* of M is the matroid with ground set L given by $M^* = \{L \setminus S \mid S \in M\}$; clearly the dual of M^* is M itself. The rank function of M^* is given by $\rho^*(S) = |S| - (\rho(L) - \rho(L \setminus S))$. The *contraction* of M to the set $L \setminus S$ is the matroid M/S with ground set $L \setminus S$ and rank function $\rho_{M/S}(X) = \rho(S \cup X) - \rho(S)$. The *restriction* of M to the set S is the matroid $M|_S$ with ground set S and independent sets $M|_S = \{I \in M \mid I \subseteq S\}$.

Proof of Lemma 1 Assume that the set

$$A = \{u \in \mathcal{U} \mid F[u] \neq D[u]\}$$

is non-empty. Let

$$u = \arg\min\{\min(F[u], D[u]) \mid u \in A\},$$

where ties are broken arbitrarily. Then $D[u] \neq F[u]$; suppose that $D[u] > F[u]$. Then for any $v \in A$, our choice of u implies that $D[v] \geq \min(D[v], F[v]) \geq \min(D[u], F[u]) = F[u]$; and for any $v \notin A$, we have $D[v] = F[v]$ by definition. In either case one of the inequalities required by condition (1) fails, so F is not maxmin-fair. Put differently, we have shown the following implication:

$$F \text{ is maxmin-fair} \implies D[u] < F[u].$$

Similarly,

$$D \text{ is maxmin-fair} \implies F[u] < D[u].$$

But then F and D cannot both be maxmin-fair. The only way out of this contradiction is to conclude that A is empty.

Proof of Theorem 1 $\implies$ Let F be maxmin-fair and consider any other distribution D. We need to show that $F \uparrow \succeq D \uparrow$ (that is, $F \uparrow$ is lexicographically largest). Define

$$A = \{u \in \mathcal{U} \mid F[u] \neq D[u]\}.$$

If A is empty, the claim is trivial; otherwise let

$$u = \arg\min\{F[u] \mid u \in A\} \text{ and } B = \{v \in \mathcal{U} \mid F[v] < F[u]\}.$$

Note that $u \in A \subseteq \overline{B}$ by our choice of u. If $D[u] > F[u]$, from the maxmin-fairness of F we infer the existence of $v \in A \subseteq \overline{B}$ such that $D[v] < F[u]$. This also holds if $D[u] < F[u]$ (then we can take $v = u$). In any case we have

$$\min\{D[v] \mid v \notin B\} < F[u] = \min\{F[v] \mid v \notin B\}$$
$$\text{and} \qquad D[v] = F[v] < F[u] \ \forall v \in B.$$

It is readily verified that this implies $F \uparrow \succ D \uparrow$.

$\impliedby$ Let F be a distribution which is not maxmin-fair. We show that F is not lexicographically largest either. Since (1) does not hold for F, there exists another distribution D and a user $u \in \mathcal{U}$ such that

$$D[u] > F[u] \text{ and } (D[v] < F[v] \implies F[v] > F[u]) \ \forall v. \tag{6}$$

For any $\varepsilon \in (0, 1)$, let X_ε denote the distribution picking F with probability $1 - \varepsilon$ and D with probability ε, so that

$$X_\varepsilon[v] = F[v] + \varepsilon(D[v] - F[v]) \quad \forall v.$$

Choose $\varepsilon > 0$ small enough so as to guarantee that

$$(F[v] < F[u] \implies X_\varepsilon[v] < X_\varepsilon[u]) \quad \forall v \tag{7}$$

and

$$(F[v] > F[u] \implies X_\varepsilon[v] > X_\varepsilon[u]) \quad \forall v. \tag{8}$$

For instance, any

$$\varepsilon < \min\left\{\frac{|F[u] - F[v]|}{|D[v] - F[v]| + |D[u] - F[u]|} \middle| F[v] \neq F[u]\right\}$$

will do. We have, by (6),

$$(F[v] \le F[u] \implies D[v] \ge F[v] \implies X_\varepsilon[v] \ge F[v]) \quad \forall v \tag{9}$$

$$\text{and} \qquad D[u] > F[u]. \tag{10}$$

But (7), (8), (9) and (10) say that $X_\varepsilon \uparrow$ is strictly larger than $F \uparrow$ in lexicographical order, as we wished to show.

The following two analogues of Lemma 1 and Theorem 1 are also needed for the proof of Theorem 2:

Lemma 7 *If F and D are both minmax-Pareto, then $F[u] = D[u]$ for all $u \in \mathcal{U}$.*

Proof Assume that the set

$$A = \{u \in \mathcal{U} \mid F[u] \ne D[u]\}$$

is non-empty. Let

$$u = \arg\max\{\max(F[u], D[u]) \mid u \in A\},$$

where ties are broken arbitrarily. Then $D[u] \ne F[u]$; suppose that $D[u] < F[u]$. Then for any $v \in A$, our choice of u implies that $D[v] \le \max(D[v], F[v]) \le \max(D[u], F[u]) = F[u]$; and for any $v \notin A$, we have $D[v] = F[v]$ by definition. In either case one of the inequalities required by the definition of minmax-Pareto efficiency fails. Put differently, we have shown the following implication:

$$F \text{ is minmax-Pareto} \implies D[u] > F[u].$$

Similarly,

$$D \text{ is minmax-Pareto} \implies F[u] > D[u].$$

But then F and D cannot both be minmax-Pareto. The only way out of this contradiction is to conclude that A is empty. □

Theorem 14 *For matroid problems, a distribution F is minmax-Pareto if and only if F is Pareto-efficient and $F \downarrow \preceq D \downarrow$ for all Pareto-efficient distributions D.*

Proof First observe that, for matroids, a distribution is Pareto-efficient if and only if it is supported over bases. For any distribution D of bases over a matroid M with ground set L, consider the distribution D^* of $(L \backslash X \mid X \sim D)$ of bases over the dual matroid M^*. Then we have $D[u] + D^*[u] = 1$ for all $u \in L$, so clearly F is minmax-Pareto if and only if F^* is maxmin-fair, which (by Theorem 1) occurs if and only if F^* is lexicographically largest for M^*, which in turn is equivalent to F being lexicographically smallest among distributions of bases of M, as we wished to show. □

Our next result asserts that the only obstruction to achieving high satisfaction probability for every user is the existence of a set of users with small rank-to-size ratio. Finding these obstruction sets will enable us to devise a divide and conquer strategy to obtain fair distributions. For instance, in Example 1 the obstruction set is given by the set of users $\{a_1, a_2, a_3\}$, which force the maximum satisfaction probability to be no larger than $\frac{2}{3}$.

Theorem 15 *Let M be a matroid with ground set L and rank function $\rho\colon 2^L \to \mathbb{N}$. The minimum satisfaction probability in a minmax-fair distribution over M is*

$$\pi(M) = \min\left\{\frac{\rho(X)}{|X|} \mid \emptyset \neq X \subseteq L\right\}.$$

Proof Any maxmin-fair distribution is supported on the collection $\mathcal{B}$ of bases of M, since extending an independent set to a base containing it never decreases any satisfaction probability. Optimizing the smallest satisfaction probability λ_1 amounts to finding a suitable distribution over $\mathcal{B}$; let us denote the corresponding probabilities by $\{p_B\}_{B\in\mathcal{B}}$. Since the probability of $v \in \mathcal{U}$ being included is $\sum_{v\in B} p_B$, maximizing the minimum such probability is modeled by Program (11) below. It may be written as a linear program by introducing an additional variable λ to be maximized, and introducing the constraints $\sum_{B\ni v} p_B \geq \lambda$. Its dual is equivalent to (12).

$$\begin{aligned} \max\ \min_{v\in\mathcal{U}} & \sum_{B\ni v} p_B \\ \text{s.t.} \quad & \sum_{B\in\mathcal{B}} p_B = 1 \\ & p_B \geq 0 \end{aligned} \tag{11}$$

$$\begin{aligned} \min\ \max_{B\in\mathcal{B}} & \sum_{v\in B} z_v \\ \text{s.t.} \quad & \sum_{v\in\mathcal{U}} z_v = 1 \\ & z_v \geq 0. \end{aligned} \tag{12}$$

Observe that $\max_{B\in\mathcal{B}} \sum_{v\in B} z_v$ is the value of a maximum-weight base of M, with weights given by $\{z_v\}_{v\in\mathcal{U}}$. Thus LP (12) encodes the task of finding an assignment of weights to elements of $\mathcal{U}$ minimizing the maximum weight of a base. We will turn this min–max problem into a pure minimization problem.

Edmonds (1971) showed that for any fixed assignment of non-negative weights to the elements of M, a maximum-weight base may be found via the greedy algorithm that examines each element in order of decreasing weight and adds it to the current set if its addition does not violate independence. Let Π denote the set of permutations of $\mathcal{U} = \{1, 2, \ldots, n\}$. Write

$$\Delta = \left\{ z \in \mathbb{R}^{\mathcal{U}} \,\middle|\, \sum_{v \in \mathcal{U}} z_v = 1, \quad z_v \geq 0 \; \forall v \in \mathcal{U} \right\}$$

for the probability simplex on $\mathcal{U}$ and let $G(\pi) = \{z \in \Delta \mid z_{\pi(1)} \geq z_{\pi(2)} \geq \cdots \geq z_{\pi(n)}\}$ denote the elements of Δ which become sorted after applying permutation $\pi \in \Pi$.

Note that if $z, z' \in G(\pi)$, then the two bases obtained via the greedy algorithm with vertex weights $\{z_v\}$ and $\{z'_v\}$ are the same. For each $\pi \in \Pi$, let $B(\pi)$ denote the base obtained via the greedy algorithm; Edmond's result may then be written as

$$\max_{B \in \mathcal{B}} \sum_{v \in B} z_v = \sum_{v \in B(\pi)} z_v \quad \text{if } z \in G(\pi).$$

By LP (12), the fairness parameter $\pi(M)$ is

$$\min_{z \in \Delta} \max_{B \in \mathcal{B}} \sum_{v \in B} z_v = \min_{\pi \in \Pi} \min_{z \in G(\pi)} \max_{B \in \mathcal{B}} \sum_{v \in B} z_v = \min_{\pi \in \Pi} \min_{z \in G(\pi)} \sum_{v \in B(\pi)} z_v. \tag{13}$$

We claim that, for each $\pi \in \Pi$ and each non-empty $X \subseteq \mathcal{U}$,

$$\min_{z \in G(\pi)} \sum_{v \in X} z_v = \min_{i \in [n]} \frac{|X \cap \pi([i])|}{i}, \tag{14}$$

where $\pi([i]) = \{\pi(1), \ldots, \pi(i)\}$.

This means that, if we are given advice on the permutation π which sorts an optimal solution z to LP (12), then we can find another solution $\hat{z}$ with the same value and whose non-zero weights are evenly distributed among the top t elements of z in sorted order, for some $t \in [n]$. For some optimal t, each of the t non-zero values of $\hat{z}_i$ is either 0 or $1/t$. To see this assuming 14, notice that we can construct such $\hat{z}$ by setting $\hat{z}_{\pi(i)} = \frac{1}{t}$ for $1 \leq i \leq t$ and $\hat{z}_j = 0$ for $j \notin \pi([t])$.

To see why (14) holds, define $d_n = z_{\pi(n)}$ and $d_i = z_{\pi(i)} - z_{\pi(i+1)} \geq 0$ for $0 < i < n$. Then $z_{\pi(i)} = \sum_{j \geq i} d_i$, hence

$$\sum_{v \in X} z_v = \sum_{i \in [n]} \left(\mathbb{1}[\pi(i) \in X] \cdot \sum_{j \geq i} d_i \right) = \sum_{j \in [n]} d_j \cdot |X \cap \pi([j])|.$$

The conditions $\sum_{v \in \mathcal{U}} z_v = 1$ and $z \in G(\pi)$ then become $\sum_i i \cdot d_i = 1$ and $d_i \geq 0$. Therefore

$$\min_{z \in G(\pi)} \sum_{v \in X} z_v = \min \left\{ \sum_{i \in [n]} d_i \cdot |X \cap \pi([i])| \,\middle|\, \sum_{i \in [n]} i \cdot d_i = 1, d_i \geq 0 \right\}.$$

The quantity in the right-hand side equals the smallest ratio (among all i) between the coefficient of d_i in the objective function ($|X \cap \pi([i])|$) and in the only equality constraint, proving (14). From (13) and (14) the theorem follows, because the greedy algorithm satisfies $|B(\pi) \cap \pi([i])| = \rho(\pi[i])$ for all i, so if $S^* = \arg\min_{S \subseteq \mathcal{U}} \frac{\rho(S)}{|S|}$, then for any $\pi \in \Pi$ we have

$$\min_{z \in G(\pi)} \sum_{v \in B(\pi)} z_v = \min_{i \in [n]} \frac{|B(\pi) \cap \pi([i])|}{i} = \min_{i \in [n]} \frac{\rho(\pi([i]))}{|\pi([i])|} \geq \frac{\rho(S^*)}{|S^*|},$$

and equality holds for any permutation where the elements of S^* precede those of $\mathcal{U} \setminus S^*$. □

Theorem 16 *Let M be a matroid with ground set L and rank function $\rho \colon 2^L \to \mathbb{N}$. The maximum satisfaction probability in a minmax-Pareto distribution over M is*

$$\Pi(M) = \max \left\{ \frac{\rho(L) - \rho(X)}{|L \setminus X|} \mid \emptyset \subseteq X \subsetneq L \right\}.$$

Proof Apply Theorem 15 to the dual matroid of M. □

An extension of Theorem 15 allows us to compute the satisfaction probability of every element of L.

Lemma 8 *Define a sequence of sets $B_1, B_2, \ldots, B_k$ iteratively by:*

$$B_i \text{ is a maximal set } X \subseteq L \setminus S_{i-1} \text{ minimizing } \frac{\rho(X \cup S_{i-1}) - \rho(S_{i-1})}{|X|},$$
$$\text{where } S_i = \bigcup_{j=1}^{i} B_j. \tag{15}$$

We stop when $S_i = L$ (which will eventually occur as the sequence $\{S_i\}$ is strictly increasing). Then for every $i, u \in B_i$, the satisfaction probability of u in a maxmin-fair distribution F is $\lambda_i = \frac{\rho(B_i)}{|B_i|}$.

(Maximality of each B_i is not required for the conclusion to hold, but its inclusion guarantees uniqueness of the sets thus defined, owing to the submodularity of ρ.)

Proof We reason by induction on the number k of sets. First, observe that Theorem 15 implies $F[u] \geq \lambda_1$ for all $u \in L$. As the expected number of satisfied elements within B_1, which obviously cannot exceed $\rho(B_1) = \lambda_1 |B_1|$, is equal to

$$\mathbb{E}_{A \sim F}[|A \cap B_1|] = \sum_{u \in B_1} F[u] \geq \lambda_1 |B_1| \tag{16}$$

by linearity of expectation, the equality $F[u] = \lambda_1$ must hold for all $u \in B_1$. If $k = 1$, this shows the result.

If $k > 1$, let D_1 be a maxmin-fair distribution for the restriction $M|_{B_1}$ of M to B_1 and let D_2 be a maxmin-fair distribution for the contraction $M/(L \setminus B_1)$ of M to the remaining elements $L \backslash B_1$. Since restriction does not change the rank function within B_1, D_1 satisfies $D_1[u] = F[u] = \lambda_1$ for all $u \in B_1$. The rank function of the contraction $M/(L\backslash B_1)$ is $\rho_{M/B_1}(X) = \rho(X \cup B_1) - \rho(B_1)$, so by applying rule (15) iteratively we obtain the same sequence of sets $B_2, \ldots, B_k$ (excluding B_1), and by the induction hypothesis D_2 satisfies $D_2[u] = \lambda_i$ for all $i \geq 2, u \in B_i$. It remains to be shown that $D_2[u] = F[u]$ for all $u \notin B_1$.

Denote by $[D_1 \cup D_2]$ the distribution of $(A \cup B \mid A \sim D_1, B \sim D_2)$. This is a distribution over independent sets of M by the following property of matroid contractions [see Lawler (1976)]:

$$\begin{aligned} &\text{for any base } B \text{ of } M|_{B_1}, \text{a subset } I \subseteq L\backslash B_1 \text{ is independent in } M/B_1 \\ &\quad \text{if and only if } I \cup B_1 \text{ is independent in } M. \end{aligned} \tag{17}$$

On the other hand, for any set A in the support of a maxmin-fair distribution F, the set $A \cap B_1$ must be a base of $M|_{B_1}$ (or else Eq. (16) would fail). Let F_2 denote the distribution $(A\backslash B_1 \mid A \sim F)$; by (17), F_2 is a distribution over elements of the contraction M/B_1.

To complete the proof, observe that $F \uparrow\succeq [D_1 \cup D_2] \uparrow$ by Theorem 1 because F is maxmin-fair. As $F[u] = D_1[u] < F[v], D[v]$ for $u \in B_1, v \in L\backslash B_1$, the fact that $F \uparrow\succeq [D_1 \cup D_2] \uparrow$ implies $F_2 \uparrow\succeq D_2 \uparrow$, and the maxmin-fairness of D_2 allows us to deduce that $F_2 \uparrow= D_2 \uparrow$. Hence, by Lemma 1, for all $v \notin B_1$ we have $F[v] = F_2[v] = D_2[v]$.

□

Similarly, we have the following for minmax-fairness.

Lemma 9 *Define a sequence of sets $B'_1, \ldots, B'_{k'}$ iteratively by:*

$$\begin{aligned} &B'_i \textit{ is a maximal set } X \subseteq S'_i \textit{ maximizing } \frac{\rho(S'_i) - \rho(S'_i\backslash X)}{|X|}, \\ &\quad \textit{where } S'_i = L\backslash \bigcup_{j=1}^{i-1} B'_j \end{aligned} \tag{18}$$

We stop when $S'_k = \emptyset$. Then for every $i, u \in B'_i$, the satisfaction probability of u in a minmax-Pareto distribution F is $\lambda'_i = \frac{\rho(B'_i)}{|B'_i|}$.

Proof We argue by induction on k. First, observe that Theorem 16 implies $F[u] \leq \lambda'_1$ for all $u \in L$. The expected number of satisfied elements within B_1 cannot be below $\lambda'_1|B_1|$ for any Pareto-efficient distribution F, otherwise we would have the contradiction

$$\rho(L) = \mathbb{E}_{A\sim F}\,|A| = \mathbb{E}_{A\sim F}[|A \cap B_1|] + \mathbb{E}_{A\sim F}[|A\backslash B_1|] < \lambda'_1|B_1| + \rho(L\backslash B_1) = \rho(L).$$

On the other hand,

$$\mathbb{E}_{A\sim F}[|A \cap B_1|] = \sum_{u \in B_1} F[u] \leq \lambda_1' |B_1| = \rho(B_1) \tag{19}$$

by linearity of expectation, so the equality $F[u] = \lambda_1'$ must hold for all $u \in B_1$. If $k = 1$, this shows the result. The rest of the proof is completely analogous to that of Lemma 8, except that we use Theorem 14 and Lemma 7 in place of Theorem 1 and Lemma 1. □

Proof of Theorem 2 It suffices to prove the equivalence (1) ⇔ (2). Inded, if it holds, then a maxmin-fair distribution simultaneously maximizes the minimum satisfaction probability and minimizes the maximum satisfaction probability (among Pareto-efficient distributions), hence it also minimizes the largest difference between two satisfaction probabilities. An easy inductive argument (omitted) shows that the equivalence (1) ⇔ (3) then follows.

To show that (1) ⇔ (2), consider the sequence $B_1, \ldots, B_k$ from Lemma 8 and the sequence $B_1', \ldots, B_{k'}'$ from Lemma 9. It suffices to show that they are the same sequence in reverse: $k = k'$ and $B_i = B_{k'+1-i}$ for all i. We proceed from top to bottom, showing by induction that for each value of i from 1 to k, $B_i' = B_{k+1-i}$. Consider any $Z \in S_{i+1} = S_i \cup B_{i+1}$, which may be split into $Z = X \cup Y$ where $X \subseteq S_i = \bigcup_{j \leq i} B_j$ and $Y \in B_{i+1}$. Then we have

$$\rho(Z) - \rho(S_i) \geq \rho(S_i \cup Y) - \rho(S_i) \geq \lambda_{i+1}|Y|,$$

where the first inequality is by submodularity of ρ, and the second by construction of B_{i+1} (15). On the other hand,

$$\rho(S_{i+1}) - \rho(S_i) = \lambda_{i+1}|B_{i+1}|,$$

hence

$$\frac{\rho(S_{i+1}) - \rho(Z)}{|S_{i+1} \backslash Z|} \leq \frac{\lambda_{i+1}|B_{i+1}| + \rho(S_i) - (\lambda_{i+1}|Y| + \rho(S_i))}{|B_{i+1} \backslash Y|} = \lambda_{i+1}.$$

Notice that equality holds when $Z = S_i$, so S_i maximizes $\frac{\rho(S_{i+1}) - \rho(Z)}{|S_{i+1} \backslash Z|}$ over all $Z \subseteq S_{i+1}$. Using (18) and the induction hypothesis, this means that $B'_{k'+1-i} = B_i$, as we wished to show.

Appendix B: Proofs for Sect. 4: a polynomial-time algorithm for maxmin-fair matching

Proof of Corollary 3 From Theorem 3 it follows that, when $\rho(L) = |R|$, the rank function of a bipartite matching problem is given by

$$\rho(S) = \min_{T \subseteq S} |\Gamma(T)| + |S| - |T|. \tag{20}$$

Define

$$\alpha = \max_{S \subsetneq L} \frac{|\Gamma(L)| - |\Gamma(S)|}{|L \setminus S|}; \qquad \beta = \max_{S \subsetneq L} \frac{\rho(L) - \rho(S)}{|L \setminus S|}.$$

In view of Theorem 16 and the equivalence between maxmin and minmax fairness for matroid problems (Theorem 2), it suffices to show that $\alpha = \beta$. Since $\rho(L) = |\Gamma(L)|$ and $\rho(S) \leq |\Gamma(S)|$ for all S, inequality $\alpha \leq \beta$ is immediate. To show that $\beta \leq \alpha$, it suffices to prove that $\rho(L) - \rho(S) - \alpha|L \setminus S| \leq 0$ for all $S \subseteq L$. This follows from (20) and the fact that $\alpha \leq \beta \leq 1$:

$$\begin{aligned} \rho(L) - \rho(S) - \alpha|L \setminus S| &= \max_{T \subseteq S} \ |\Gamma(L)| - |\Gamma(T)| - |S| + |T| - \alpha|L \setminus S| \\ &\leq \max_{T \subseteq S} \ \alpha|L \setminus T| - |S \setminus T| - \alpha|L \setminus S| \\ &\leq 0. \end{aligned}$$

Lemma 10 *Define a sequence of sets $B_1, B_2, \ldots, B_k$ iteratively by:*

$$\begin{aligned} &B_i \text{ is a maximal set } X \subseteq L \setminus S_{i-1} \text{ minimizing } \frac{|\Gamma(X \cup S_{i-1})| - |\Gamma(S_i)|}{|X|}, \\ &\quad \text{where } S_i = \bigcup\nolimits_{j<i} B_i. \end{aligned} \tag{21}$$

Stop when $S_k = L$. Then for every $i, u \in B_i$, the satisfaction probability of u in a maxmin-fair distribution F is $\lambda_i = \frac{|\Gamma(B_i)|}{|B_i|}$, and any $w \in \Gamma(B_i) \setminus \Gamma(S_{i-1})$ is matched to some $u \in B_i$ with probability 1.

Proof Since the sequence $S_0, S_1, \ldots$ is strictly increasing (with respect to inclusion) and L is finite, there exists some k such that $S_k = L$.

For each $i = 0, \ldots, k$, let H_i denote the graph $(G/S_{i-1})|_{L \setminus S_{i-1}}$, i.e., the result of removing the vertices in S_{i-1} and all their incident edges. For $i = 1, \ldots, k$, we argue by induction on i that the coverage probabilities of F outside of S_{i-1} coincide with those of a maxmin-fair distribution for H_i; and and moreover the probabilities are as prescribed by the statement of the lemma.

The case $i = 1$ is trivial, so assume $i > 1$. By Corollary 2, there is a distribution of matchings in H_i with minimum satisfaction probability at least λ_i; the expected number of covered elements from B_i is then at least $\lambda_i |B_i| = |\Gamma(B_i) \setminus \Gamma(S_{i-1})| = |\Gamma_{H_i}(B_i)|$. Hence equality must always hold, and the maxmin-fair distribution F_i for H_i has satisfaction probability precisely λ_i for all $u \in B_i$. By the induction hypothesis, $F[u] = F_i[u] = \lambda_i$ for all $u \in B_i$. Now observe that the neighbors of B_i that belong to S_{i-1} are already matched with probability 1. There are only $|\Gamma_{H_i}(B_i)|$ other neighbors, and since the expected number of covered neighbours of B_i in F is equal to $|\Gamma_{H_i}(B_i)|$, it follows that any $w \in \Gamma_{H_i}(B_i)$ is matched to some $v \in B_i$ with probability 1 in F. In particular, in F no element of $\Gamma_{H_i}(B_i)$ is matched to any vertex outside B_i with non-zero probability, so the satisfaction probabilities of F outside of S_i must coincide with those of of a maxmin-fair distribution for H_{i+1}. □

Lemma 11 *For any two distinct fairly isolated sets X and Y, either $X \subseteq Y$ or $Y \subseteq X$ holds.*

Proof By Corollaries 2 and 3, $X \neq \emptyset$ is fairly isolated if and only if $X = L$ or

$$\Pi(G|_X) = \max_{S \subsetneq X} \frac{|\Gamma(X)| - |\Gamma(S)|}{|X \setminus S|} < \pi(G/X) = \min_{T \supsetneq X} \frac{|\Gamma(T)| - |\Gamma(X)|}{|T \setminus X|}.$$

For any two sets A, B such that $A \subsetneq B$, define $d(A \mid B) = \frac{|\Gamma(A \cup B)| - |\Gamma(B)|}{|A \setminus B|}$. Then we can rewrite the definition of fair isolation (including the case $X = L$) as:

$$X \text{ is fairly isolated} \quad \Leftrightarrow \quad d(X \mid S) < d(T \mid X) \quad \forall\, S \subsetneq X, T \supsetneq X.$$

Now assume for contradiction X and Y are fairly isolated but $X \setminus Y$ and $Y \setminus X$ are both non-empty. Then $d(X \mid X \cap Y)$ and $d(Y \mid X \cap Y)$ are well defined; assume without loss of generality that $d(X \mid X \cap Y) \leq d(Y \mid X \cap Y)$. Then

$$d(X \mid X \cap Y) \leq d(Y \mid X \cap Y) < d(X \cup Y \mid Y),$$

where we used the fair isolation of Y. But this contradicts the submodularity of Γ. □

Proof of Theorem 5 Let $B'_1, \ldots, B'_k$ be the sequence of sets given by Lemma 10 and define $S'_i = \bigcup_{j \leq i} B'_i$, $\lambda'_i = \frac{\Gamma(S'_i \cup B'_i) - \Gamma(S'_i)}{|B'_i|}$ and $\lambda'_0 = 0$. We show that $S'_1, \ldots, S'_k$ comprise all fairly isolated sets. Assuming this for the moment, notice that by definition these sets form a chain, and the sets $B'_1, \ldots, B'_k$ satisfy property (a) by Lemma 10. Part (b) follows then by applying the fair decomposition to the dual of the matching matroid, using Corollary 3, and recalling that maxmin-fairness and minmax-Pareto efficiency are equivalent for matroids (Theorem 2).

To see that $S'_1, \ldots, S'_{k-1}$ are fairly isolated, notice that $S'_k = L$ indeed is by definition, whereas for $i < k$ we have $\Pi(G|_{S'_i}) = \max_{u \in S'_i} F[u] = \lambda'_i < \lambda'_{i+1} = \pi(G/S'_i)$. This meets the definition of fair separation from Sect. 4.4.

The fact that the fairly isolated sets form a chain is a direct consequence of Lemma 11. Finally, assume for contradiction that some fairly isolated set X exists other than $S'_1, \ldots, S'_k$. Then $S'_i \subsetneq X \subsetneq S'_{i+1}$ for some i, $0 \leq i < k$. Then we have

$$\lambda'_{i+1} \leq \frac{|\Gamma(X)| - |\Gamma(S'_i)|}{|X \setminus S'_i|} < \frac{|\Gamma(S'_{i+1})| - |\Gamma(X)|}{|S'_{i+1} \setminus X|}, \tag{22}$$

where the first inequality is by construction of B'_{i+1} and S'_{i+1}, and the second by the fair isolation of X. But then

$$\lambda'_{i+1}|S'_{i+1} \setminus X| + |\Gamma(X)| < |\Gamma(S'_{i+1})| = |\Gamma(S'_i)| + \lambda'_{i+1}|B'_i|,$$

i.e.,

$$|\Gamma(X)| - |\Gamma(S'_i)| < \lambda'_{i+1}|X \setminus S'_i|,$$

contradicting (22).

References

Big data: Seizing opportunities, preserving values. Technical report, US Executive Office of the President (2014). https://obamawhitehouse.archives.gov/sites/default/files/docs/big_data_privacy_report_may_1_2014.pdf. Accessed 21 Jan 2019
Big data: A report on algorithmic systems, opportunity, and civil rights. Technical report, US Executive Office of the President (2016). https://obamawhitehouse.archives.gov/blog/2016/05/04/big-risks-big-opportunities-intersection-big-data-and-civil-rights. Accessed 21 Jan 2019
Bansal N, Sviridenko M (2006) The Santa Claus problem. In: Proceedings of the thirty-eighth annual ACM symposium on theory of computing, STOC '06. ACM, New York, NY, USA, pp 31–40. https://doi.org/10.1145/1132516.1132522
Berge C (1957) Two theorems in graph theory. PNAS 43(9):842–844
Bertsekas DP, Gallager RG, Humblet P (1992) Data networks, vol 2. Prentice-Hall International, Upper Saddle River
Bertsimas D, Farias VF, Trichakis N (2011) The price of fairness. Oper Res 59(1):17–31
Birkhoff G (1946) Tres observaciones sobre el álgebra lineal. Univ Nac Tucumán Rev Ser A 5:147–151
Bogomolnaia A, Moulin H (2001) A new solution to the random assignment problem. J Econ Theory 100(2):295–328
Bogomolnaia A, Moulin H (2004) Random matching under dichotomous preferences. Econometrica 72(1):257–279
Brams SJ, Jones MA, Klamler C et al (2006) Better ways to cut a cake. Notices AMS 53(11):1314–1321
Brânzei S, Gkatzelis V, Mehta R (2017) Nash social welfare approximation for strategic agents. In: EC. ACM, pp 611–628
Brown JR (1979) The sharing problem. Oper Res 27(2):324–340
Caragiannis I, Kurokawa D, Moulin H, Procaccia AD, Shah N, Wang J (2016) The unreasonable fairness of maximum Nash welfare. In: EC
Charikar M (2000) Greedy approximation algorithms for finding dense components in a graph. In: APPROX, pp 84–95
Cheng CT (2010) Understanding the generalized median stable matchings. Algorithmica 58(1):34–51
Cherkassky BV, Goldberg AV (1997) On implementing the push-relabel method for the maximum flow problem. Algorithmica 19(4):390–410
Cherkassky BV, Goldberg AV (2004) HIPR 3.5. Technical report, IG systems, Inc. http://www.avglab.com/andrew/soft.html. Accessed 21 Jan 2019
Cole R, Ost K, Schirra S (2001) Edge-coloring bipartite multigraphs in O (E log D) time. Combinatorica 21(1):5–12
Coluccia A, D'Alconzo A, Ricciato F (2012) On the optimality of max–min fairness in resource allocation. Ann Telecommun 67(1–2):15–26
Corbett-Davies S, Pierson E, Feller A, Goel S, Huq A (2017) Algorithmic decision making and the cost of fairness. In: KDD
Dulmage AL, Mendelsohn NS (1958) Coverings of bipartite graphs. Can JRL Math 10(4):516–534
Dwork C (2017) What's fair? In: KDD [Keynote]
Dwork C, Hardt M, Pitassi T, Reingold O, Zemel R (2012) Fairness through awareness. In: Proceedings of the 3rd conference on innovations in theoretical computer science. ACM, pp 214–226
Edmonds J (1965) Paths, trees, and flowers. Can JRL Math 17(3):449–467
Edmonds J (1971) Matroids and the greedy algorithm. Math Prog 1(1):127–136
Edmonds J, Fulkerson D (1965) Transversals and matroid partition. J Res Nat Bur Stand 69B:147–153
Edmonds J, Pruhs K (2006) Cake cutting really is not a piece of cake. In: SODA. Society for Industrial and Applied Mathematics, pp 271–278
Feldman M, Friedler SA, Moeller J, Scheidegger C, Venkatasubramanian S (2015) Certifying and removing disparate impact. In: KDD
Gale D, Shapley LS (1962) College admissions and the stability of marriage. Am Math Mon 69(1):9–15
Gallo G, Grigoriadis MD, Tarjan RE (1989) A fast parametric maximum flow algorithm and applications. SIAM J Comput 18(1):30–55
Goel A, Kapralov M, Khanna S (2013) Perfect matchings in $O(n \log n)$ time in regular bipartite graphs. SIAM J Comput 42(3):1392–1404
Goldberg AV, Rao S (1998) Beyond the flow decomposition barrier. J ACM 45(5):783–797

Goldberg AV, Tarjan RE (1988) A new approach to the maximum-flow problem. J ACM (JACM) 35(4):921–940
Hall P (1935) On representatives of subsets. JRL Lond Math Soc 1(1):26–30
Hardy GH, Littlewood JE, Pólya G (1952) Inequalities. CUP
Heidari H, Ferrari C, Gummadi KP, Krause A (2018) Fairness behind a veil of ignorance: a welfare analysis for automated decision making. In: Neural and information processing systems (NeurIPS)
Heidari H, Loi M, Gummadi KP, Krause A (2019) A moral framework for understanding of fair ml through economic models of equality of opportunity. In: ACM conference on fairness, accountability, and transparency (ACM FAT*)
Ichimori T, Ishii H, Nishida T (1982) Optimal sharing. Math Program 23(1):341–348
Joseph M, Kearns M, Morgenstern JH, Roth A (2016) Fairness in learning: classic and contextual bandits. In: Advances in neural information processing systems, pp 325–333
Kamada Y, Kojima F (2015) Efficient matching under distributional constraints: theory and applications. Am Econ Rev 105(1):67–99
Kamiran F, Calders T (2011) Data preprocessing techniques for classification without discrimination. Knowl Inf Syst 33(1):1–33
Katoh N, Ibaraki T, Mine H (1985) An algorithm for the equipollent resource allocation problem. Math Oper Res 10(1):44–53
Kearns M (2017) Fair algorithms for machine learning. In: EC
Klamler C (2010) Fair division. In: Kilgour DM, Eden C (eds) Handbook of group decision and negotiation. Springer, Berlin, pp 183–202
Lawler EL (1976) Combinatorial optimization: networks and matroids. Courier Corporation, North Chelmsford
Lovász L, Plummer M (2009) Matching theory. AMS Chelsea Publishing Series, North-Holland
McElfresh DC, Dickerson JP (2018) Balancing lexicographic fairness and a utilitarian objective with application to kidney exchange. In: Thirty-second AAAI conference on artificial intelligence
Megiddo N (1977) A good algorithm for lexicographically optimal flows in multi-terminal networks. Bull Am Math Soc 83(3):407–409
Pedreschi D, Ruggieri S, Turini F (2008) Discrimination-aware data mining. In: KDD
Picard JC, Queyranne M (1980) On the structure of all minimum cuts in a network and applications. In: Rayward-Smith VJ (ed) Combinatorial optimization II. Mathematical programming studies, vol 13. Springer, Berlin, Heidelberg
Rawls J (1971) A theory of justice. Harvard University Press, Cambridge
Roth AE, Sönmez T, Ünver MU (2005) Pairwise kidney exchange. J Econ Theory 125(2):151–188
Teo CP, Sethuraman J (1998) The geometry of fractional stable matchings and its applications. Math Oper Res 23(4):874–891
Triesch E (1992) A short proof that matching matroids are transversal. Appli Math Lett 5(2):19–20
Zliobaite I, Kamiran F, Calders T (2011) Handling conditional discrimination. In: ICDM

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

David García-Soriano[1] · Francesco Bonchi[1,2]

Francesco Bonchi
francesco.bonchi@isi.it

1 ISI Foundation, Turin, Italy

2 Eurecat, Barcelona, Spain