Matroids, Matchings, and Fairness

Flavio Chierichetti
Dipartimento di Informatica
Sapienza University
Rome, Italy

Ravi Kumar
Google Research
1600 Amphitheater Parkway
Mountain View, CA 94043

Silvio Lattanzi
Google Research
Brandschenkestrasse 110
Zurich, ZH 8002

Sergei Vassilvitskii
Google Research
76 9th Ave
New York, NY 10011

Abstract

The need for fairness in machine learning algorithms is increasingly critical. A recent focus has been on developing fair versions of classical algorithms, such as those for bandit learning, regression, and clustering. We extend this line of work to include algorithms for optimization subject to one or multiple matroid constraints. We map out this problem space, showing optimal solutions, approximation algorithms, or hardness results depending on the specific problem flavor. Our algorithms are efficient and empirical experiments demonstrate that fairness is achievable without a large compromise to the overall objective.

1 Introduction

The desire to use machine learning to assist in human decision making has spawned a large area of research in understanding the impact of such systems not only on the society as a whole, but also the specific impact on different subpopulations [Pleiss et al., 2017, Zafar et al., 2017, Corbett-Davies et al., 2017]. Fairness is a critical requirement of such systems. Recent research has shown that while there are several natural ways to quantify the fairness of a particular system, no one of them is universal, and except for trivial cases, satisfying one means violating another [Pleiss et al., 2017, Corbett-Davies et al., 2017, Kleinberg et al., 2017].

In parallel to understanding the interplay between different definitions of fairness, researchers have looked to develop algorithms for finding fair solutions to specific classes of problems. For example, there is recent work on fair regression [Joseph et al., 2016], fair ranking [Celis et al., 2018c], fair clustering [Chierichetti et al., 2017, Rösner and Schmidt, 2018], fair bandit algorithms [Liu et al., 2017], and many others.

In this work we tackle another large class of problems that form a useful primitive in machine learning and optimization, that of optimizing a set function subject to a set of matroid constraints. A matroid is a combinatorial object that generalizes linear independence between vectors and is general enough to encode cardinality constraints (e.g., selecting at most k elements), connectivity constraints (e.g., selecting a spanning tree of a graph), or matching constraints (ensuring a subgraph has a perfect matching).

What makes matroid constraints popular is that they are general enough to encode many different types of problems yet relatively easy to optimize over. For instance, a simple greedy algorithm optimizes a modular function subject to a single matroid constraint, and the celebrated algorithm by Edmonds [1970] shows how to optimize a function subject to two matroid constraints simultaneously. While the problem becomes NP-hard when there are three ore more matroids involved, there is an algorithm that finds a $1/(k-1)$-approximation to finding the largest set that satisfies k matroid constraints (see Section 2 for more details).

Thanks to the expressive power of matroids, by designing new algorithms for matroids and intersection of matroids optimization under balance constraints, we automatically obtain many new algorithms for practical problems under matching, cardinality, or connectivity constraints. A nice example that has many practical applications is fair matching: matching students to schools, ad slots to advertisers, search results to ranked positions, etc. In all of these settings we have natural matching constraints (school capacity, advertiser budget, single result per position), edge weights representing the quality of the match (student's aptitude for a specific school, relevance of ad to user, likelihood of result click), and properties of edges that require system-wide balance (making sure not too many students need bussing, ads from all political parties are represented, diversity in results).

Our contributions. Since optimization with matroid con-

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

straints is an important primitive, we ask if the optimization can be done in a fair manner. As there is no universal metric of fairness, we adopt the definition used in previous work by Celis et al. [2018a,b,c], Chierichetti et al. [2017], Rösner and Schmidt [2018] and ask for a *balanced* solution.

As an example, consider an e-commerce site that presents item reviews to customers. Since any popular item will have hundreds of reviews we may want to ensure that the reviews shown to a particular user are balanced. This balance may be epitomized in that the reviews include both recent purchasers and long-time users, or that they include both positive and negative opinions, or that they include reviews from different countries around the world. Generally, balance means that in each relevant dimension there are limits on how frequently each feature value is present. We make the setting precise in Section 2.

If we only insist on upper bounds, i.e., ensuring no single feature value dominates, then we show how to encode the balance constraint as an additional matroid constraint, and use known algorithms to achieve an optimal solution (Lemma 2).

If, on the other hand, we insist that all values are represented (i.e., both lower and upper bounds on the appearance of each feature value), the situation is harder. When the number of distinct values is a constant, c, we show how to find optimal balanced sets subject to a single matroid constraint (Lemma 3). In the special case when $c = 2$, we further show a $2/3$-approximate solution when there are two matroid constraints that must be satisfied simultaneously (Corollary 9). Finally, we give some evidence of the hardness of selecting a balanced set subject to two matroid constraints by drawing on connections to the Exact Matching problem, which is not known to be in P, and to the 1-in-3SAT problem, which is known to be hard under the Exponential Time Hypothesis (ETH). More precisely, we show in Theorem 10 that for unboundedly many colors the problem is NP-hard and, under the ETH, it requires time exponential in the number of colors.

Finally we give some empirical evidence that simply ignoring balance when looking for an optimal solution can lead to highly unbalanced results. While this is not surprising, we also show that taking balance into account does not dramatically degrade the value of the solution, even when only approximately optimal solutions can be found (Section 5).

Related work. In recent years there has been a lot of attention to fairness in machine learning, along two directions.

The first is trying to understand what it means for an algorithm to be fair. Examples of this line of work are the results on statistical parity [Luong et al., 2011, Kamishima et al., 2011], disparate impact [Feldman et al., 2015], and individual fairness [Dwork et al., 2012]. More recent works by Corbett-Davies et al. [2017] and Kleinberg et al. [2017] show that some of the desired properties of fairness may be incompatible with each other; see the recent work of Selbst et al. [2019] for a different perspective.

A second line of work focuses on algorithm design to achieve fair outcomes. Here the bulk of the work has been on supervised [Luong et al., 2011, Hardt et al., 2016] and online learning [Joseph et al., 2016]. A similar direction to the one studied in our work is that of learning intermediate representations that are fair by design, see for example the work by Zemel et al. [2013] and Kamishima et al. [2011]. However, unlike their work, here we focus on designing algorithms with provable guarantees for a wide range of problems under balance constraints.

In this paper we adopt the notion of fairness known as *disparate impact* introduced by Feldman et al. [2015]. This notion is close to the notion of $p\%$-rule as a measure for fairness. That is a generalization of the 80%-rule advocated by US Equal Employment Opportunity Commission [Biddle, 2006]. We note here that the vast majority of previous definitions, such as statistical parity and equality of opportunity, are in the traditional learning setting. There one want to prove the classifier is going to be "fair" over items drawn from an example distribution, hence the guarantees are probabilistic over that distribution. However, there are many natural settings, especially in combinatorial optimization, where there is no distribution per se and we want to ensure a fair allocation on *every* instance. For example, the questions of candidate selection in fair rankings [Celis et al., 2018c], fair clustering [Chierichetti et al., 2017, Rösner and Schmidt, 2018], fair voting [Celis et al., 2018a], are of this type. All of the works above employ the same notion of balance that we explore in this paper.

The works closer to ours are the ones on fair ranking and fair voting. Celis et al. [2018c] study the problem of fair ranking and to do so they focus their attention to a special case of the fair matching problem. In particular the problem they solve in their paper is a special case of our setting, for two reasons: (i) they address weighted bipartite matching, which is a special case of the our problem of optimizing a function subject to multiple matroid constraints and (ii) they make additional assumptions about structure of the weights (related to their goal of fairness in ranking), making their result even more specialized. In fact, they do not have an algorithm for general weights, even in the limited setting of unweighted bipartite matchings, whereas we give a constant-factor approximation for the same problem[1]. Celis et al. [2018a] study fair multiwinner voting scheme and they approach the problem as a submodular optimization problem under a single matroid constraint. The results in their paper do not apply to intersection of matroids and so their result cannot be used to obtain algorithms in our setting.

[1]Nevertheless it is important to note that in their setting they can prove the existence of a polynomial time algorithm.

Finally fair optimization is loosely related to diversity optimization, although the connection is largely superficial. There are several algorithms for diversity maximization under matroids constraints but they do not imply results in our setting, e.g., Lin and Bilmes [2011] consider adding knapsack or covering constraints to submodular functions to use them for document summarization. These types of constraints are incomparable to balance constraints, as they cannot capture the notion of fairness that we address. Similarly Ahmed et al. [2017] add diversity to weighted bipartite matchings, using a quadratic program and a greedy approach. Unfortunately also in this case, the diversity notion cannot capture the balance constraints we wish to enforce.

2 Preliminaries

In this work we consider the question of maximizing a set function subject to matroid and balance constraints.

A matroid is a generalization of the notion of linear independence in vector spaces with many applications in geometry, topology, economics, combinatorial optimization, network theory, and coding theory. There are many equivalent definitions of matroids; we choose the following one.

A finite *matroid* M is a pair $(E, \mathcal{I})$, where E is a finite set (called the *ground set*) and $\mathcal{I}$ is a family of subsets of E, called the *independent sets*, with the following properties:

- $\emptyset \in \mathcal{I}$;
- (hereditary property) if $A \in \mathcal{I}$ and $A' \subseteq A$ then $A' \in \mathcal{I}$; and
- (exchange property) if $A, B \in \mathcal{I}$ and $|B| < |A|$, then $\exists x \in A \setminus B$ such that $B \cup \{x\} \in \mathcal{I}$.

The *intersection* of two or more matroids is the family of sets that are simultaneously independent in all the matroids.

Computationally speaking, the problem of finding the maximum independent set in one matroid or in the intersection of two matroids can be solved in polynomial time. To solve the former, simply greedily add elements that satisfy the constraint; solving the latter problem is far from obvious and follows from a beautiful result by Edmonds [1970]. The problem is NP-complete for three or more matroids; however, there is a $\frac{1}{k-1+\epsilon}$-approximation algorithm for finding the (weighted) maximum independent set in the intersection of k matroids [Lee et al., 2010].

In this work we add fairness constraints to the general problem of matroid optimization. Following previous work by Celis et al. [2018a,b,c], Chierichetti et al. [2017], and Rösner and Schmidt [2018], we encode fairness by posing additional *balance* constraints on the solution. We assume that every element $e \in E$ has a color. We denote by C the set of colors, and by $c : E \to C$ the assignment of colors to the elements. For every subset $S \subseteq E$ of elements, we define $S_{c'}$ as the subset of elements with color $c' \in C$, i.e., $S_{c'} = \{e \in S \mid c(e) = c'\}$.

We first define a notion that captures the amount of color imbalance in an independent set.

Definition 1 ((α, β)-balance)**.** *For $0 \leq \alpha \leq \beta \leq 1$, an independent set S is* (α, β)-balanced *if for every color $c \in C$, it holds that*

$$\alpha \leq \frac{|S_c|}{|S|} \leq \beta.$$

Thus a set is (α, β)-balanced if every color occurs in at least an α fraction of the elements and in at most a β fraction. Our goal is to find a maximum size independent set subject to both matroid and balance constraints. Formally, a *γ-approximation algorithm* will output an (α, β)-balanced independent set of size at least $\gamma \cdot \text{OPT}$, where OPT is the value of the optimum solution. Two cases of (α, β)-balance are particularly interesting: $\alpha = \beta = 1/|C|$, i.e., all colors occur equally often in S; in this case, we say that S is *perfectly balanced*. Also, the case where (ii) $\alpha = 0$, i.e., where no color occurs more than $\beta|S|$ times: here, we say that S is *β-limited*.

3 Warmup: Single matroid

In this section we consider the balanced matroid optimization problem subject to a single matroid constraint with an arbitrary number of colors, $|C|$. Recall that without the additional balance consideration, the simple greedy algorithm is guaranteed to yield the optimum solution. Here we first show that the β-limiting constraint can be encoded as an additional matroid constraint, and then use this observation to give a polynomial time algorithm to find the optimum (α, β)-balanced solution.

Recall the notion of a partition matroid. Let $\mathcal{T} = \{T_c\}_{c \in C}$ be a collection of disjoint sets that partition E and let d_c be integers such that $0 \leq d_c \leq |T_c|$ for all c. Then the family $\mathcal{I}$ of independent sets for a *partition matroid* is such that if $I \in \mathcal{I}$ then $\forall c$, $|I \cap T_c| \leq d_c$. Suppose we are given a generic matroid constraint. To encode an extra β-limiting constraint, we first guess the size s of the optimal solution (i.e., the set of largest size that is independent for the matroid constraint and that satisfies the β-limiting constraint), and then create a partition matroid that allows at most $d_c = \lfloor \beta \cdot s \rfloor$ elements for each color c. By our guess on s, the matroid intersection algorithm [Edmonds, 1970] will obtain a solution. While the largest feasible s is unknown a priori, the matroid intersection algorithm can be tried for each $s \in [n]$.

Lemma 2. *There exists a polynomial time algorithm that finds a largest independent set of a matroid, subject to an additional β-limiting constraint.*

We now extend this observation to arbitrary (α, β)-balanced sets with a single matroid constraint.

Lemma 3. *For any $\alpha < \beta$ and constant number of colors, there is a polynomial time algorithm to find an (α, β)-balanced maximum independent set subject to a single matroid constraint.*

Proof. Since $|C|$, the number of colors, is constant, we can guess the number of elements of each color in the optimal solution (there are $O(n^{|C|})$ such guesses). For one such guess, say, $n_1, \ldots, n_{|C|}$ elements per color class, we can add a partition matroid that bounds the number of elements of color i by n_i, for each $i \in [C]$, as in Lemma 2. The matroid intersection algorithm will find the optimal solution for the given guess of $n_1, \ldots, n_{|C|}$. We can then return the optimal solution among all of those satisfying the (α, β)-balance constraints. Observe that the optimal solution to the initial problem is tight for one of the partition matroids that would have been considered. □

Since finding a maximum *weighted* independent set subject to the intersection of two matroids is still in P we get the following corollary.

Corollary 4. *For any $\alpha < \beta$ and constant number of colors, there is a polynomial time algorithm to find an (α, β)-balanced maximum weighted independent set subject to a single matroid constraint.*

Unfortunately, this simple approach cannot be extended to find balanced maximum independent sets subject to intersection of two or more matroids. In this case the direct algorithm only yields an approximately optimal solution, and thus we have no guarantees that the result would remain balanced. In the next section we show a much more nuanced approach for finding fair maximum independent sets subject to intersection of matroids.

4 Intersection of two matroids

In this section we show a $2/3$-approximation algorithm for finding (α, β)-balanced maximum independent sets, when $|C| = 2$. This directly implies approximation algorithms for the balanced bipartite matching, b-matching, and many other problems. We will then discuss the hardness of finding the exact solution in Section 4.1.

For simplicity of exposition, we let the color set be $C = \{\text{RED}, \text{BLUE}\}$. Before describing our positive result we recall some basic notions from Edmonds [1970]. We start with the concept of an exchange graph.

Definition 5 (Exchange graph)**.** *Let $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ be two matroids on the same ground set and let $T \in \mathcal{I}_1 \cap \mathcal{I}_2$. Then the* exchange graph $D_{M_1, M_2}(T)$ *is the directed bipartite graph with bipartition T and $E \setminus T$ such that (y, x) is an arc if $T - y + x \in \mathcal{I}_1$ and (x, y) is an arc if $T - y + x \in \mathcal{I}_2$.*

Let $X_1 = \{x \notin T \mid T + x \in \mathcal{I}_1\}$ be the set of *sources* and $X_2 = \{x \notin T \mid T + x \in \mathcal{I}_2\}$ be the set of *sinks*. Edmonds proved that if there is no *augmenting path*[2] between elements in X_1 and X_2, then T is a set of maximum size in $\mathcal{I}_1 \cap \mathcal{I}_2$. Otherwise, it is possible to increase the size of T by finding a simple augmenting path from X_1 to X_2.

The main idea in our algorithm is to extend this approach for intersection of two matroids to incorporate the balance condition. (For simplicity of exposition we first present the algorithm that finds a perfectly balanced set, and then show how to adapt it to work for any (α, β)-balance requirement.) We begin by finding a maximum independent set T using Edmonds algorithm. If T is balanced, then we are done. Otherwise, we rebalance it in such a way that its size does not decrease too much. We now explain the details of the rebalancing step.

Without of loss generality, let us assume that there are more BLUE elements than RED elements in T. We compute a maximum independent set, T', in the intersection of the two matroids on the ground set reduced to its RED elements: $M_1^{\text{RED}} = (E_{\text{RED}}, \mathcal{I}_1)$ and $M_2^{\text{RED}} = (E_{\text{RED}}, \mathcal{I}_2)$ where E_{RED} is the set of RED elements in E.

We now use T' to iteratively change T. Every iteration will make T more balanced, at the cost of potentially reducing its size. We will then show that the algorithm terminates (i.e., T becomes balanced) while the set T is still large compared to the optimal solution.

First, define two auxiliary matroids $M_1' = (T' \cup T, \mathcal{I}_1)$ and $M_2' = (T' \cup T, \mathcal{I}_2)$ with the same independent sets but different ground set. Then construct the exchange graph $D_{M_1', M_2'}(T)$ and define the *augmented source* set as $X_1^+ = \{x \notin T \mid T + x \in \mathcal{I}_1 \text{ or } T + x - y \in \mathcal{I}_1 \text{ for some BLUE } y \in T\}$ and the *augmented sink* set as $X_2^+ = \{x \notin T \mid T + x \in \mathcal{I}_2 \text{ or } T + x - y \in \mathcal{I}_2 \text{ for some BLUE } y \in T\}$. Observe that X_1^+ is composed of two types of nodes, those classified as sources by the standard Edmonds algorithm (i.e., $\{x \notin T \mid T + x \in \mathcal{I}_1\}$), and those that exchange a BLUE node in T for a RED node not in T: $\{x \notin T, y \in T, c(y) = \text{BLUE} \mid T + x - y \in \mathcal{I}_1\}$. We call the second set BLUE *sources* and call the node $y \in T$ such that $T + x - y \in \mathcal{I}_1$ the *balancer* of x. Similarly, for sinks, we can partition them into those classified as sinks by the standard Edmonds algorithm, and those that exchange a BLUE node in T for a RED node not in T. Again, we call the corresponding BLUE node a balancer.

We then find the shortest augmenting path between a source element in X_1^+ and a sink element in X_2^+. Let P be such an augmenting path. We replace T with $T \triangle P$. If the source or the sink node had a BLUE balancer, we remove them

[2] An *augmenting path* between a source s and sink t is a path P in the exchange graph starting and finishing in $E \setminus T$. Note that that $T \triangle (P \setminus \{s, t\})$ is independent in both matroids M_1 and M_2; here, $\triangle$ denotes the symmetric difference.

from T. Observe that in this way T remains an independent set for both M_1 and M_2. Moreover, with every augmentation the number of RED elements increases by one, and the number of BLUE elements decreases by at most two.

We iterate this process until one of the following three conditions is met: (i) the set T is perfectly balanced, (ii) the number of BLUE elements in T exceeds the number of RED elements in T by at most two: $|T_{\text{BLUE}}| \leq |T_{\text{RED}}| + 2$, or (iii) there are no more augmenting paths. In the first case we return T, in the second and third case we remove arbitrary BLUE elements from T until it is balanced. (Note by the hereditary property, this maintains the feasibility of T.)

Let $\mathsf{Edmonds}(E, \mathcal{I}_1, \mathcal{I}_2)$ denote the output of Edmonds algorithm to compute the maximum independent set in the intersection $\mathcal{I}_1 \cap \mathcal{I}_2$ using the ground set E. We present the formal pseudocode in Algorithm 1.

Algorithm 1 Approximation algorithm for intersection of two matroids in 2-colors balanced setting

Input: Two matroids $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$
Output: An approximate balanced maximum independent set for $\mathcal{I}_1 \cap \mathcal{I}_2$

1: $T \leftarrow \mathsf{Edmonds}(E, \mathcal{I}_1, \mathcal{I}_2)$
2: (Assume w.l.o.g. that $|T_{\text{RED}}| \leq |T_{\text{BLUE}}|$; the other case is symmetric)
3: $T' \leftarrow \mathsf{Edmonds}(E_{\text{RED}}, \mathcal{I}, \mathcal{I}_2)$
4: (Start the rebalancing phase)
5: **while** T still changes **do**
6: **if** T is (α, β)-balanced **then**
7: Return T
8: **if** $|T_{\text{BLUE}}| - |T_{\text{RED}}| \leq 2$ **then**
9: **break**
10: Let $M_1' = (T' \cup T, \mathcal{I}_1)$, $M_2' = (T' \cup T, \mathcal{I}_2)$.
11: Let $X_1^+ = \{x \notin T \mid T + x \in \mathcal{I}_1\} \cup \{x \notin T, y \in T, c(y) = \text{BLUE} \mid T + x - y \in \mathcal{I}_1\}$
12: Let $X_2^+ = \{x \notin T \mid T + x \in \mathcal{I}_2\} \cup \{x \notin T, y \in T, c(y) = \text{BLUE} \mid T + x - y \in \mathcal{I}_2\}$
13: Construct the exchange graph $D_{M_1', M_2'}(T)$ and find the shortest augmenting path P between X_1^+ and X_2^+
14: **if** $P \neq \emptyset$ **then**
15: $T \leftarrow T \triangle P$
16: Remove from T the balancers associated the sources and sinks in P.
17: **while** T is not (α, β)-balanced **do**
18: Remove one BLUE element from T
19: Return T

We next prove guarantees on the approximation achieved by this algorithm. Our proof follows the same outline as Nomikos et al. [2007]. In a sense, we generalize that approach to handle the case of fair maximum balanced independent sets in intersection of matroids. We start by proving some some useful statements.

Lemma 6. *Steps 15 and 16 of Algorithm 1 increase the number of* RED *elements in T by exactly one and decrease the number of* BLUE *elements in T at most by two. Furthermore after Step 16, $T \in \mathcal{I}_1 \cap \mathcal{I}_2$.*

Proof. First note that the shortest augmenting path P contains only RED elements and the number of elements in $|(T' \setminus T) \cap P| = |T \cap P| + 1$ so $T \triangle P$ contains one more RED element than T. Furthermore in Step 16 we delete at most two BLUE elements: the balancers associated with the source and the sink.

Finally, note that for every element in $x \in P \setminus T$ added to T we remove from T some element forming a cycle with it in $\mathcal{I}_1$ and $\mathcal{I}_2$ therefore the new set is feasible. □

Lemma 7. *If there is no augmenting path between X_1^+ and X_2^+, then T has the same number of* RED *elements as the optimal solution.*

Proof. Consider two auxiliary matroids with the same ground set but with different independent sets, $M_1'' = (T' \cup T_{\text{RED}}, \mathcal{I}_1)$ and $M_2'' = (T' \cup T_{\text{RED}}, \mathcal{I}_2)$, where T_{RED} is the set of RED elements in T. First, note that the maximum independent set in the intersection of the two matroids is of size $|T'|$ by definition of T' as the maximum set among all RED elements.

Second, consider the exchange graph $D_{M_1'', M_2''}(T)$, the source set (i.e., $X_1 = \{x \notin T_{\text{RED}} \mid T_{\text{RED}} + x \in \mathcal{I}_1\}$), and the sink set (i.e., $X_2 = \{x \notin T_{\text{RED}} \mid T_{\text{RED}} + x \in \mathcal{I}_2\}$) in Edmonds algorithm. Note that in $D_{M_1'', M_2''}(T)$ there is no augmenting path between X_1 and X_2 otherwise there will be an augmenting path between X_1^+ and X_2^+. So T_{RED} is a maximum independent set, thus $|T_{\text{RED}}| = |T'|$. So the final solution contains the maximum possible number of RED elements in an independent set. □

Using these, we show the approximation guarantee of the algorithm. Let ALG be the size of perfectly balanced solution returned by the algorithm and let OPT be the size of the perfectly balanced optimal solution.

Theorem 8. $\text{ALG} \geq (2/3) \cdot \text{OPT} - 2$.

Proof. Let r and b denote the number of RED and BLUE elements in the solution T after step 2 (before the rebalancing phase); we have assumed without loss of generality that $b > r$. Since we first looked for the maximum independent set without considering the balance constraint, $\text{OPT} \leq r + b$.

First, note that during the execution of the algorithm the number of BLUE elements is always higher or equal to the number of RED elements.

Let $t = \lfloor \frac{b-r}{3} \rfloor$ be the number of iterations. Since the number of BLUE elements decreases by at most two every iteration, the number of BLUE elements remaining is at least

$$b - 2t = b - 2\left\lfloor \frac{b-r}{3} \right\rfloor \geq \frac{b+2r}{3}.$$

The number of RED elements increases by one every iteration, hence the number of RED elements remaining is

$$r + t = r + \left\lfloor \frac{b-r}{3} \right\rfloor \leq \frac{b+2r}{3}.$$

Since the number of BLUE elements decreases monotonically, the only way the algorithm has terminated prior to iteration t is if no augmenting path P was found. In this case, by Lemma 7 the number of RED elements is the same as in the optimal solution, and the algorithm returns an optimal solution after further pruning some BLUE elements, if necessary, in lines 17 and 18.

Otherwise, consider what happens if rebalancing stops after t iterations. Then the total size of the balanced solution would be at least

$$2(r+t) \geq 2\left(\frac{b+2r}{3} - 1\right) \geq \frac{2}{3}(b+r) - 2 \geq \frac{2}{3}\text{OPT} - 2.$$

Since the algorithm will run for $t' \geq t$ iterations, the solution at the end will be of size

$$\text{ALG} = 2(r+t') \geq 2(r+t) \geq \frac{2}{3}\text{OPT} - 2. \quad \square$$

We note that Algorithm 1 leads to the same approximation ratio for any (α, β)-balanced constraint. Notably, we stop rebalancing when both the colors are present in at least an α fraction of the elements. We then remove individual elements to satisfy the upper bound constraints.

Corollary 9. *The modified algorithm returns a solution* ALG *with* $\text{ALG} \geq 2/3 \cdot \text{OPT} - 3$ *to the* (α, β)*-balanced maximum independent set in* $\mathcal{I}_1 \cap \mathcal{I}_2$.

Proof. Let w be the number of elements of the dominant color in any optimal solution. Let $s = |T'|$, b (resp., r) be the number of BLUE (resp., RED) elements in the maximum independent set computed in step 1, OPT the value of the optimum solution and ALG the value of our solution. Define $z = \min\{s, w, b + r - w\}$; note that $\text{OPT} \leq w + z$.

First, during the execution of the algorithm the number of BLUE elements is always higher than the number of RED elements. Furthermore if we exit the loop because there are no more augmenting paths we return an optimal solution because the number of RED elements is optimal.

So for the rest of the proof we can assume that we exit the loop only because T is (α, β)-balanced or because $|T_{\text{BLUE}}| - |T_{\text{RED}}| \leq 2$.

During the rebalancing phase if the number of BLUE elements is at least $w + 2$, then we have that the number of elements in T is larger than $w + 2 + r + \frac{b-w-2}{2} \geq w + r + \frac{b-w}{2} + 1$. Therefore, if T becomes (α, β)-balanced while the number of BLUE elements is bigger or equal than $w+2$, then we have $\text{ALG} \geq w + \frac{b+r-w}{2} + \frac{r}{2} + 1 \geq w + \frac{z}{2} + 1$. Since $\text{OPT} \leq w + z$, we get $\text{OPT} \leq \text{ALG} + \frac{z}{2}$ and by $z \leq w$ we get $\text{ALG} \geq \frac{3}{2}z$. Thus in this case we get $\text{OPT} \leq \frac{4}{3}\text{ALG}$.

Furthermore, if we have $|T_{\text{BLUE}}| - |T_{\text{RED}}| \leq 2$ and the number of BLUE elements is bigger or equal than $w + 2$, then we can just remove one or two BLUE elements and return an (α, β)-balanced solution with $2w$ elements. So in this case $\text{ALG} \geq \text{OPT}$.

We now focus on the case where the number of BLUE elements is smaller than $w + 2$ when we exit the loop. Note that until the number of BLUE elements is bigger or equal than $w+2$, we have $|T| \geq w + \frac{z}{2} + 1$. So the first time when the number of BLUE elements is smaller than $w+2$, we cannot have that $|T| \geq w + \frac{z}{2}$ and furthermore $|T_{\text{RED}}| \geq \frac{z}{2} - 1$ and $|T_{\text{BLUE}}| \leq w + 1$. So we cannot decrease the size of T by more than $\lceil \frac{w-1-(z/2-1)}{3} \rceil$ during the execution of the main loop. This is true because in every iteration in which we decrease the size of T we substitute two BLUE elements with one RED element and hence after $\lceil \frac{w-1-(z/2-1)}{3} \rceil$ such iterations we have $|T_{\text{BLUE}}| - |T_{\text{RED}}| \leq 2$. Thus at the end of the loop we have $\text{ALG} \geq \frac{2}{3}w + \frac{2}{3}z - 1$. Since $\text{OPT} \leq w + z$, we get $\frac{2}{3}\text{OPT} - 1 \leq \text{ALG}$.

Finally note that if the main loop terminates because $|T_{\text{BLUE}}| - |T_{\text{RED}}| \leq 2$, one can just remove one or two BLUE elements and became (α, β)-balanced so we get $\text{ALG} \geq \frac{2}{3}\text{OPT} - 3$. $\square$

To prove a bound on the computational complexity of the algorithm, we assume to have access to an oracle with running time $O(Q)$ to check whether a set is independent in a given matroid (for many families of matroids such a check can be implemented in constant time)[3]. Then, the running time of the algorithm is $O(Qn^3)$, where $n = |E|$ is the size of the ground set. To construct an exchange graph we need to check at most n^2 possible edges. Furthermore, for any exchange graph we can find the shortest augmenting path in time $O(n^2)$. Since, we construct at most n exchange graphs, and that we search for one augmenting path in each of them, we obtain the bound on the running time.

4.1 Hardness of balanced optimization

In the previous section we gave an approximation algorithm when considering intersection of two matroids. Here, we discuss the the computational hardness of an instantiation of that scenario, namely the maximum matching problem, and show that while it is easy for two colors, it is likely to be hard when the number of colors is allowed to grow. (Recall that the problem is NP-hard for the intersection of three or more matroids [Lee et al., 2010].)

[3]The worst-case time to check whether a set is independent depends on the matroid and how it is represented. For example, for trees, it involves checking if a set of edges contains a cycle, which can be done in linear time. For bipartite matching, it involves checking if each node is incident to at most one edge, which again takes linear time. For cardinality and partition matroids, the checks are trivial and can be done in constant time.

The (α, β)-balanced bipartite matching problem with a generic number $|C|$ of colors can be solved exactly in (randomized) time $|V|^{O(|C|)}$. To obtain such a result, one could use a recent algorithm of Czabarka et al. [2018]. This randomized algorithm solves the $m_1, \ldots, m_c$ exact edge-matching problem: given a graph $G(V, E)$, with E partitioned into c color classes $E_1, \ldots, E_c$, and given integers $m_1, \ldots, m_c$, does there exist a matching $M \subseteq E$ in G such that $|M \cap E_i| = m_i$ for each $i \in [c]$? The algorithm runs in time $|V|^{O(c)}$ and it is therefore a randomized polynomial time algorithm so long as the number of colors is constant.

It is easy to see that the algorithm of Czabarka et al. [2018] can be used to solve the (α, β)-balanced bipartite matching problem with c colors in randomized polynomial time: one can just iterate across all the possible color counts (there are less than $|V|^c$ of them) that satisfy the given (α, β)-balance property, and check whether that color count can be realized by a matching. The total running time is then $|V|^{O(c)}$, and the algorithm needs to be randomized.[4]

We also show that as the number of colors grows, the running time has to degenerate into an exponential form, under the Exponential Time Hypothesis (ETH).[5]

Theorem 10. *Under the ETH, the time needed to solve the balanced bipartite matching problem is $2^{\Omega(|C|)}$, where C is the set of colors.*

Proof. Plaisted and Zaks [1980] give a reduction that, starting from a 1-in-3SAT instance on n variables and m clauses, produces a bipartite graph with $O(n+m)$ vertices, $O(n+m)$ edges with $O(n+m)$ colors: the formula admits a satisfying assignment iff one can find a perfect matching in the bipartite graph containing at most one edge per color. The reduction in, e.g., Schaefer [1978] starting from a 3SAT instance on n variables and m clauses, produces a 1-in-3SAT instance on $O(n+m)$ variables and $O(n+m)$ clauses. Therefore, there exists a reduction from 3SAT on n variables and m clauses to the $(0, \frac{3}{|V|})$-balanced bipartite matching problem on the graph $G(V, E)$ (i.e., does there exist a $(0, \frac{3}{|V|})$-balanced bipartite matching of value $\geq |V|/2$)? Each such matching must contain each color at most once, with $|V| = O(n+m)$, $|E| = O(n+m)$.

Under the ETH, 3SAT on n variables and $m = \Theta(n)$ clauses requires time $2^{\Omega(n)}$ to be solved. It follows that, under the ETH, the balanced bipartite matching problem requires time $2^{\Omega(|C|)}$. □

[4]The randomization requirement follows from the connection with the $m_1, \ldots, m_c$ exact edge-matching problem: it is still unknown whether this problem can be solved in deterministic polynomial time. A deterministic polynomial time algorithm is not known to exist even in the case where $c = 2$ and $m_1 + m_2 = |V|/2$, i.e., in the perfect matching case: this problem was introduced by Papadimitriou and Yannakakis [1982] under the name of "Exact Matching" and a randomized polynomial time algorithm for it was shown to exist by Mulmuley et al. [1987].

[5] I.e., under the assumption that the minimum running time required for solving a generic SAT instance on n variables is $2^{\Omega(n)}$ [Impagliazzo and Paturi, 2001].

5 Experiments

In this section we give a brief view into the performance of our algorithms on real-world data. Since these are the first algorithms that allow for optimization subject to a balance constraint, our goal is to look at the difference between the optimum solution for the unbalanced and balanced cases. While the optimum balanced solution will necessarily be no larger than the optimum unbalanced one, we would like to study the loss in the objective if one seeks a balanced solution. For simplicity we focus on the perfectly balanced bipartite matching problem.

Datasets. We consider three datasets: two from the SNAP repository (`snap.stanford.edu`) and one from MovieLens (`grouplens.org/datasets/`).

Amazon reviews. This dataset (AMAZON) is obtained by processing review information about 548,552 different products on Amazon. From this dataset we generate a bipartite graph between users and products where edges represent reviews. We label negative (one or two star) reviews as RED and all remaining reviews as BLUE. In this way we obtain 860,650 RED edges and 5,498,535 BLUE edges for a total of 6,359,185 edges.

Wikipedia election data. This dataset (WIKI) represents vote history for administrator elections, coming from nearly 2,800 elections. We construct a bipartite graph between voters and nominees where every edge either represents a positive, negative, or neutral vote. In our experiments we label positive votes RED, and neutral and negative votes BLUE. In this way we obtain 30,091 RED edges and 83,949 BLUE edges for a total of 114,040 edges.

MovieLens ratings. This dataset (MOVIELENS) represents 20,000,263 movie ratings by 138,493 users on a 5-star scale with half-star increments. The bipartite graph is between the users and 26,744 movies where each edge represents a rating. To convert this graph into two colors, we mark the edges with ratings below 3.5 as RED and the others as BLUE. This way we obtain 9,995,410 RED edges and 10,004,853 BLUE edges.

Algorithms. We look for the maximum matching in these bipartite graphs. Note that a matching can be encoded as an intersection of two matroids (one making sure that the degree of nodes on the left bipartition is at most one, the other making sure that the degree of nodes on the right bipartition is at most one).

We compute the optimal matching using the Hungarian method, which does not take edge color into account. We then implement a slightly simpler version of the balanced matroid partition algorithm. This algorithm works by (i)

finding the respective maximum matching on RED and BLUE edges, (ii) considering the union of these two matchings, which will be a set of paths and cycles, and (iii) optimally solving the fair matching problem on the paths and cycles, which turns out to be easy. This simple algorithm is guaranteed to give a $1/2$-approximation (details in the Supplementary Material). To contrast our work, we also consider the following *greedy* baseline: construct a maximal matching greedily, alternately adding BLUE and RED edges. We use the term *coverage* to denote the ratio of the perfectly balanced matching computed by our algorithm to the (possibly unbalanced) maximal matching. For consistency we always take the fraction of the RED edges in the maximal matching to be a measure of the imbalance.

Main resuls: Imbalance and coverage. For the AMAZON dataset, the maximum matching (size 360,636) is very unbalanced with only 11% being RED. On the other hand, when insisting on a balanced matching, we can select 108,570 edges, yielding over 30.1% coverage. For the WIKI dataset, in the maximum matching (size 2,389) over 60% of edges are RED. On the other hand we can find a balanced matching of size 1,928, representing over 80.7% coverage. For the MOVIELENS dataset, the maximum matching (size 19,010) has 44% RED edges. On the other hand, we can find a balanced matching of size 16,814, representing over 88.4% coverage. We also find that the greedy baseline achieves ≈94% of our solution on the WIKI and MOVIELENS datasets.

The empirical findings conform to our intuition: without controlling for balance the optimum solution can be very unbalanced, adding the additional balance constraint is easy and recovers a non-trivial fraction of the optimum.

Stability. To see how stable the results are with respect to a given dataset, we sample subgraphs of different sizes in a dataset and consider the imbalance in the maximal matching and the coverage of our algorithm. Figure 1 shows the results for AMAZON and MOVIELENS. It is clear that while the coverage of our algorithm and the imbalance of the maximal matching are at odds with each other, the absolute numbers are largely stable as a function of the subgraph size. This suggests that these properties are inherent to the underlying dataset, which makes the balance requirements all the more interesting and important.

Role of imbalance. Our final experiment concerns the interplay between imbalance and coverage, when the underlying bipartite graph structure is fixed. In other words, the edges remain the same, but the colors can change. To this end, we consider the MOVIELENS dataset and modify the threshold of when an edge is labeled RED; thus the color of an edge changes from RED to BLUE as the threshold increases. Figure 2 shows the results. The imbalance falls monotonically as one would expect (since the threshold value increases, fewer edges will be colored RED) but interestingly, the coverage exhibits a unimodal behavior, maximized when the threshold value is around 3.5; this is the value for which the MOVIELENS graph has roughly the same number of BLUE and RED edges. Investigating this phenomenon in detail in a more formal setting is an interesting avenue for research.

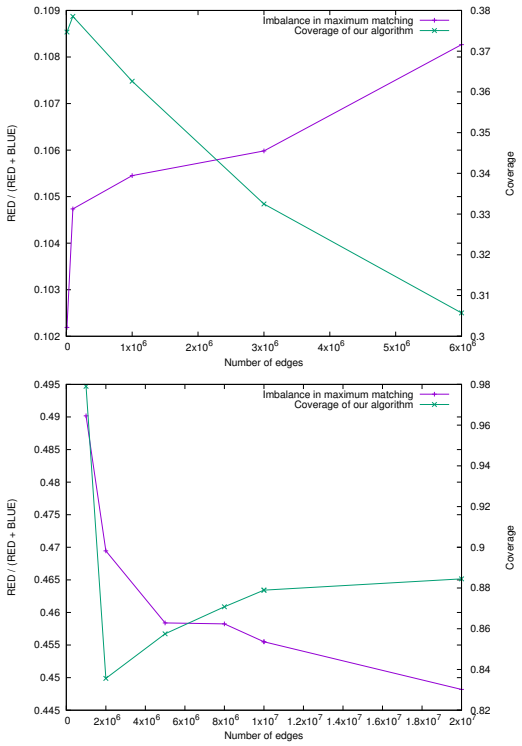

Figure 1: Imbalance of the maximal matching and the coverage of our algorithm for AMAZON and MOVIELENS.

6 Conclusions

In this paper we build upon the work of fair machine learning algorithms, and extend the notion of balanced solution sets to the question of optimization subject to matroid constraints. While the problem is computationally harder than its unbalanced variant, we give efficient approximation algorithms, and empirically demonstrate in some cases perfectly balanced solutions can be nearly optimal, as long as we optimize with balance in mind.

Acknowledgments

Flavio Chierichetti was supported in part by the ERC Starting Grant DMAP 680153, by a Google Focused Research Award, by the "Dipartimenti di Eccellenza 2018-2022" grant awarded to the Dipartimento di Informatica at Sapienza, and by the Bertinoro International Center for Informatics.

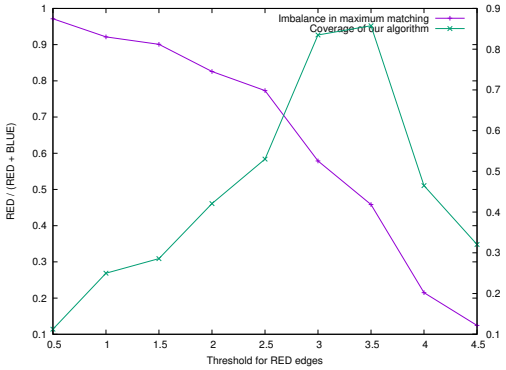

Figure 2: Imbalance and coverage as a function of rating threshold for MOVIELENS.

References

Faez Ahmed, John P. Dickerson, and Mark Fuge. Diverse weighted bipartite b-matching. In *IJCAI*, pages 35–41, 2017.

Dan Biddle. *Adverse Impact and Test Validation: A Practitioner's Guide to Valid and Defensible Employment Testing*. Gower Publishing, Ltd., 2006.

L. Elisa Celis, Lingxiao Huang, and Nisheeth K. Vishnoi. Multiwinner voting with fairness constraints. In *IJCAI*, pages 144–151, 2018a.

L. Elisa Celis, Vijay Keswani, Damian Straszak, Amit Deshpande, Tarun Kathuria, and Nisheeth K. Vishnoi. Fair and diverse DPP-based data summarization. In *ICML*, pages 715–724, 2018b.

L. Elisa Celis, Damian Straszak, and Nisheeth K. Vishnoi. Ranking with fairness constraints. In *ICALP*, pages 28:1–28:15, 2018c.

Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. In *NIPS*, pages 5036–5044, 2017.

Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic decision making and the cost of fairness. In *KDD*, pages 797–806, 2017.

Eva Czabarka, Laszlo A. Szekely, Zoltan Toroczkai, and Shanise Walker. An algebraic Monte-Carlo algorithm for the partition adjacency matrix realization problem. Technical Report 1708.08242v2, arXiv, 2018.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *ITCS*, pages 214–226, 2012.

Jack Edmonds. Submodular functions, matroids, and certain polyhedra. In *Combinatorial Structures and their Applications (Proc. 1969 Calgary Conference)*, pages 69–87. Gordon and Breach, 1970.

Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *KDD*, pages 259–268, 2015.

Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *NIPS*, pages 3315–3323, 2016.

Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *JCSS*, 62(2):367–375, 2001.

Matthew Joseph, Michael Kearns, Jamie H. Morgenstern, and Aaron Roth. Fairness in learning: Classic and contextual bandits. In *NIPS*, pages 325–333, 2016.

Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *ICDM Workshops*, pages 643–650, 2011.

Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. In *ITCS*, pages 43:1–43:23, 2017.

Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Math. OR*, 35(4):795–806, 2010.

Hui Lin and Jeff A. Bilmes. A class of submodular functions for document summarization. In *ACL-HLT*, pages 510–520, 2011.

Yang Liu, Goran Radanovic, Christos Dimitrakakis, Debmalya Mandal, and David Parkes. Calibrated fairness in bandits. In *FAT-ML at KDD*, 2017.

Binh Thanh Luong, Salvatore Ruggieri, and Franco Turini. k-NN as an implementation of situation testing for discrimination discovery and prevention. In *KDD*, pages 502–510, 2011.

Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987.

Christos Nomikos, Aris Pagourtzis, and Stathis Zachos. Randomized and approximation algorithms for blue-red matching. In *MFCS*, pages 715–725, 2007.

Christos H. Papadimitriou and Mihalis Yannakakis. The complexity of restricted spanning tree problems. *JACM*, 29(2):285–309, 1982.

David A. Plaisted and Samuel Zaks. An NP-complete matching problem. *Discrete Applied Mathematics*, 2(1): 65–72, 1980.

Geoff Pleiss, Manish Raghavan, Felix Wu, Jon M. Kleinberg, and Kilian Q. Weinberger. On fairness and calibration. In *NIPS*, pages 5684–5693, 2017.

Clemens Rösner and Melanie Schmidt. Privacy preserving clustering with constraints. In *ICALP*, pages 96:1–96:14, 2018.

Thomas J. Schaefer. The complexity of satisfiability problems. In *STOC*, pages 216–226, 1978.

Andrew D. Selbst, Danah Boyd, Sorelle A. Friedler, Suresh Venkatasubramanian, and Janet Vertesi. Fairness and abstraction in sociotechnical systems. In *FAT**, pages 59–68, 2019.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, and Krishna P. Gummadi. Fairness constraints: Mechanisms for fair classification. In *AISTATS*, pages 259–268, 2017.

Richard S. Zemel, Yu Wu, Kevin Swersky, Toniann Pitassi, and Cynthia Dwork. Learning fair representations. In *ICML*, pages 325–333, 2013.