Information Processing and Management 59 (2022) 102707

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/ipm

Fair Top-*k* Ranking with multiple protected groups

Meike Zehlike [a,*], Tom Sühr [b], Ricardo Baeza-Yates [c], Francesco Bonchi [d,e], Carlos Castillo [f], Sara Hajian [g]

[a] *Humboldt Universität zu Berlin, Max–Planck-Inst. for Software Systems, Saarbrücken, Germany*
[b] *Technische Universität Berlin, Germany*
[c] *Inst. for Experiential AI, Northeastern University, CA, USA*
[d] *ISI Foundation, Turin, Italy*
[e] *Eurecat, Barcelona, Spain*
[f] *Universitat Pompeu Fabra, Barcelona, Spain*
[g] *Nets Group, Copenhagen, Denmark*

ARTICLE INFO

Keywords:
Group fairness, ranking
Algorithmic discrimination
Post processing
Multinomial distribution
Algorithmic fairness

ABSTRACT

Ranking items or people is a fundamental operation at the basis of several processes and services, not all of them happening online. Ranking is required for different tasks, including search, personalization, recommendation, and filtering. While traditionally ranking has been aimed solely at maximizing some global utility function, recently the awareness of potential discrimination for some of the elements to rank, has captured the attention of researchers, which have thus started devising ranking systems which are non-discriminatory or *fair* for the items being ranked. So far, researchers have mostly focused on *group fairness*, which is usually expressed in the form of constraints on the fraction of elements from some protected groups that should be included in the top-k positions, for any relevant k. These constraints are needed in order to correct implicit societal biases existing in the input data and reflected in the relevance or fitness score computed.

In this article, we tackle the problem of selecting a subset of k individuals from a pool of $n \gg k$ candidates, maximizing global utility (*i.e.*, selecting the "best" candidates) while respecting given group-fairness criteria. In particular, to tackle this *Fair Top-k Ranking* problem, we adopt a ranked group-fairness definition which extends the standard notion of group fairness based on protected groups, by ensuring that the proportion of protected candidates in every prefix of the top-k ranking remains statistically above, or indistinguishable from, a given minimum threshold. Our notion of utility requires, intuitively, that every individual included in the top-k should be more qualified than every candidate not included; and that for every pair of candidates in the top-k, the more qualified candidate should be ranked above.

The main contribution of this paper is an algorithm for producing a fair top-k ranking that can be used when more than one protected group is present, which means that a statistical test based on a multinomial distribution needs to be used instead of one for a binomial distribution, as the original FA*IR algorithms does. This poses important technical challenges and increases both the space and time complexity of the re-ranking algorithm. Our experimental assessment on real-world datasets shows that our approach yields small distortions with respect to rankings that maximize utility without considering our fairness criteria.

* Corresponding author.
E-mail addresses: meikezehlike@mpi-sws.org (M. Zehlike), tom.suehr@googlemail.com (T. Sühr), rbaeza@acm.org (R. Baeza-Yates), francesco.bonchi@isi.it (F. Bonchi), chato@acm.org (C. Castillo), sara.hajian@gmail.com (S. Hajian).

https://doi.org/10.1016/j.ipm.2021.102707
Received 6 January 2021; Received in revised form 29 June 2021; Accepted 22 July 2021
Available online 24 September 2021

1. Introduction

Ranking is a fundamental task at the basis of several processes and services in the online and offline world, including search, recommendation, personalization, and filtering. The position that an item (or a person, as in our case of interest) receives in a ranking can have a substantial economic impact. For instance, in a university admission only the top-k ranked applicants are admitted (for some specific value of k), so that those ones not making it into the top-k have zero benefit. In online search platforms, the position in a ranking influences to a large extent the attention that an item receives: only the top-k items are shown in the first results page, due to physical limitations of the screen. Moreover, it is well known that users inspecting the results are susceptible to *position bias*, *i.e.*, they pay attention mostly to the top of the page and thus to the top-ranked items (Craswell, Zoeter, Taylor, & Ramsey, 2008). As people search engines are increasingly common for job recruiting (Raghavan, Barocas, Kleinberg, & Levy, 2020) and even for finding companionship or friendship, top-k ranking algorithms might have a direct and tangible impact on people's life.

The main concern motivating this work is that ranking algorithms based in machine learning, may produce ranked lists that can systematically reduce the visibility of already disadvantaged groups (Dwork, Hardt, Pitassi, Reingold, & Zemel, 2012; Pedreshi, Ruggieri, & Turini, 2008) and legally protected categories such as people with disabilities, racial or ethnic minorities, or an under-represented gender in a specific professional domain. This systematic bias may have various origins, including training data annotated by biased experts, click data from users exhibiting various kinds of biases, and even differences in document construction across groups (*e.g.*, women and men complete sections of their professional profiles differently (Altenburger, De, Frazier, Avteniev, & Hamilton, 2017)). Furthermore, it is assumed that this bias manifests differently across groups, rendering inter-group relevance scores incomparable with each other.

According to Friedman and Nissenbaum (1996), a computer system is *biased* "if it systematically and unfairly discriminate[s] against certain individuals or groups of individuals in favor of others. A system discriminates unfairly if it denies an opportunity or a good, or if it assigns an undesirable outcome to an individual or a group of individuals on grounds that are unreasonable or inappropriate". Yet "unfair discrimination alone does not give rise to bias unless it occurs systematically" and "systematic discrimination does not establish bias unless it is joined with an unfair outcome". In a ranking, the desired outcome for an individual is to be ranked as high as possible. In the cases in which there is a complete or big drop in utility after the kth position, the desired outcome is to be ranked amongst the top-k. A ranking outcome can be deemed unfair, if members of one or more protected groups are systematically ranked below those of a privileged group. In this regard, the main issue is that machine learning models trained on datasets incorporating *preexisting societal biases* will learn and structure such biases, eventually producing biased results, which once deployed in decision making in the real world, can reinforce existing biases and strengthen inequalities (O'Neil, 2016). In some cases, the bias can even be amplified (Kleinberg, Lakkaraju, Leskovec, Ludwig, & Mullainathan, 2018).

Based on this observation, in this article we study the problem of producing a ranking that we will consider fair, given legally-protected attributes. Specifically, we extend the top-k ranking algorithm FA*IR (Zehlike et al., 2017) and its *ranked group fairness criterion* to handle multiple protected groups. Intuitively, given a set G of different minority groups, our aim is to produce a ranking in which the proportion of each group $g \in G$ in any ranking prefix does not fall below given minimum proportions p_G, where p_G is a vector containing the respective minimum proportion p_g per group g. In this ranking, we also would like to preserve relevance/utility as much as possible. A formal definition of the problem can be found in Section 3.

The input vector p_G of minimum proportions can originate from a legal mandate or from voluntary affirmative actions. For instance, the US Equal Employment Opportunity Commission sets a goal of 12% of workers with disabilities in federal agencies in the US,[1] while in Spain, a minimum of 40% of political candidates in voting districts exceeding a certain size must be women (Verge, 2010). In other cases, such quotas might be adopted voluntarily, for instance through a diversity charter.[2] In general these measures do not mandate perfect parity, in part because distributions of qualifications across groups can be unbalanced for legitimate, explainable reasons (Pedreschi, Ruggieri, & Turini, 2009a; Žliobaite, Kamiran, & Calders, 2011). The ranked group fairness criterion compares the number of protected elements in every prefix of the ranking with the expected number of protected elements if they were picked at random using a multinomially distributed statistical process ("dice rolls" with each side g of the dice representing a group, and having success probability $p_g \in p_G$). Given that we use a statistical test for this comparison, we include a significance parameter α corresponding to the probability of a type-I-error, which means rejecting a fair ranking.

Example. In the following we want to illustrate why we have to extend the Fair Top-k Ranking Problem from Zehlike et al. (2017) to be based on a multinomial distribution, instead of just applying binomial FA*IR to a ranking for each group consecutively. Consider the rankings in Table 1, corresponding to five hypothetical ways of ranking ten candidate applicants in a credit approval process. The rankings are obtained based on the credit worthiness of each applicant, taking into account different features such as account status, credit duration, and credit amount. We also present to which protected group each individual belongs based on their demographics "age" and "race". Suppose we have two protected groups, namely "young" white candidates and "old" black candidates, and one non-protected group, old white candidates; for now, in this example, we do not consider young black candidates on purpose.

Running FA*IR for one protected group (Zehlike et al., 2017) with minimum proportion $p = 0.3$ and significance $\alpha = 0.1$ translates into requiring at least one protected candidate in the top seven positions of a ranking with ten positions in total. Suppose we set the same minimum proportions for both protected groups $p_{\text{y/w}} = p_{\text{o/b}} = 0.3$ and we assume that the colorblind ranking (upper most ranking in Table 1) does not meet ranked group fairness with these requirements. The next two rankings have been obtained using FA*IR for one protected group, each focusing either on the protected group of old blacks or young whites, while treating the rest

[1] US EEOC: https://www1.eeoc.gov/eeoc/newsroom/release/1-3-17.cfm, Jan 2017.

[2] European Commission: http://ec.europa.eu/justice/discrimination/diversity/charters/.

Table 1
Five possible rankings of top-10 results with people from two protected groups: young white (red) and old black persons (blue), and one non-protected group: old white persons. Suppose we set the same minimum proportions for these two groups $p_{y/w} = p_{o/b} = 0.3$. The first ranking shows the colorblind ranking, which is unfair for both groups (note that the remaining protected candidates have been ranked outside of the top-10 and are not shown). The next three rankings are created using FA*IR from (Zehlike et al., 2017) with respective minimum proportions: The second ranking would be considered fair for young people, but not for black people; the third ranking would be considered fair for black people, but not for young people; the fourth ranking combines black and young people into one protected group, but because the group-specific bias is higher for young people, they are ranked below all black candidates; the fifth ranking was created using the multinomial extension of FA*IR and would satisfy the multinomial ranked group fairness condition for both young and black people. Note that this is only one possible fair ranking out of many possibilities.

Position	1	2	3	4	5	6	7	8	9	10
Colorblind	o/w	o/w	o/w	o/w	o/w	o/w	o/w	o/w	o/b	y/w
Young whites $p_{y/w} = 0.3$	o/w	o/w	o/w	o/w	o/w	o/w	y/w	o/w	o/w	o/b
Old blacks $p_{o/b} = 0.3$	o/w	o/w	o/w	o/w	o/w	o/w	o/b	o/w	o/w	y/w
y/w and o/b as one group $p_{y/w+o/b} = 0.6$;	o/w	o/w	o/b	o/w	o/b	o/w	o/b	o/w	y/w	o/w
y/w and o/b as distinct groups $p_{y/w} = p_{o/b} = 0.3$;	o/w	o/w	y/w	o/b	o/w	y/w	o/b	o/w	y/w	o/b

as non-protected. We see directly why this is problematic: equal minimum proportions p result in exactly the same requirements of protected candidates for each group. It is not clear whether an old black or a young white candidate should be preferred at position 7. More importantly, it is also not clear how to proceed with the candidate that is not chosen. Let us assume that the young white candidate is chosen to be ranked to position 7. As old black candidates are not considered as protected in this setting, then they would be ranked even lower than their original position in the colorblind ranking (observe the second ranking in Table 1).

To overcome this, suppose we aggregate all protected groups together into a single protected group and add up the minimum proportions to one that fits the total share of protected candidates. In our example, we combine the group of young whites and old blacks into one group with minimum proportion $p = 0.6$. This however, neglects to account for any bias that manifests differently across protected groups. Hence, if the bias in credit scores of young people is significantly higher than for black people, then all young whites will be ranked below the old blacks in the fair ranking (observe the fourth ranking in Table 1). Additionally, combining all protected groups into one group ignores the problem of *intersectional discrimination*. Intersectional discrimination refers to the fact that personal, political, and social identities of an individual can be combined to form a unique profile that can be discriminated. If we also considered young black people as a third protected group, the bias in their scores would manifest itself through two dimensions of discrimination, namely through age *and* race, which may lead to even more bias in their scores, and thus potentially to the lowest positions, even when applying FA*IR. Therefore young black candidates should be considered as their own group whose scores are not comparable with the ones of young whites, or old blacks.

From the above we conclude that providing a fairness definition for rankings that either treats protected groups consecutively, or assembles all groups into one big group, cannot guarantee a fair ranking for multiple protected groups with different bias manifestations. This highlights the need for a ranked group fairness notion for multiple protected groups that is based on a multinomial distribution. The fifth ranking in Table 1 is an example of a multinomially fair top-10 ranking for the protected groups young and black. It is created using our *multinomial ranked group fairness criteria* and the multinomial FA*IR algorithm proposed in this paper. Note that this is only one of the possible fair rankings out of many possibilities. We remark that our method is suitable for concerns about intersectional discrimination, by forming groups through the Cartesian product of protected attributes.

1.1. Contributions and roadmap

We define and analyze the Fair Top-k Ranking problem with multiple protected groups, in which we want to determine a subset of k candidates from a large pool of $n \gg k$ candidates, in a way that maintains high utility (selects the "best" candidates from each group), subject to a group fairness criterion. This work addresses multiple protected groups extending our previous work (Zehlike et al., 2017) that only considered the single protected group case.

Our notion of utility assumes that we want to invite the most qualified candidates from each group, while their qualification is equal to a relevance score calculated by a ranking algorithm. This score is assumed to be based on relevant metrics for evaluating candidates for a work position, which depending on the specific skills required for the job could be their grades (*e.g.*, Grade Point Average), their results in a standardized knowledge/skills test specific for a job, such as their typing speed in words per minute for typists, or their number of hours of flight in the case of pilots. We note that this measurement will embody *preexisting bias* (*e.g.*, if black pilots are given less opportunities to pilot an airplane they will accumulate less flight hours), as well as *technical bias*, as learning algorithms are known to be susceptible to direct and indirect discrimination (Hajian, Bonchi, & Castillo, 2016; Hajian & Domingo-Ferrer, 2013). We furthermore note that different manifestations of such bias exist for each group and are usually stronger for intersectional groups, that is, the preexisting bias against black women is stronger than the one for women or blacks in general.

Our utility principle is operationalized in two ways. First, we prefer rankings in which every individual in the top-k is more qualified than every candidate not included in the top-k (or in which the difference in their qualifications is small). We call this criterion *selection utility*. Second, we prefer rankings in which for every pair of individuals included in the top-k, either the more qualified candidate is ranked above, or the difference in their qualifications is small. We call this criterion *ordering utility*. Note however, that in a setting with multiple protected groups, optimal selection and ordering utility cannot be guaranteed because of

said differences in the group skews. Mathematically this means that the optimal solution for multinomial ranked group fairness (*i.e.*, for more than one protected group) is a solution space rather than just a single point as it was in (Zehlike et al., 2017), while the optimal solution in terms of utility is still a single point within said solution space whose location depends on the candidate set at hand. We want to stress that trying to find this point of optimal utility corresponds to a world view in which one assumes that utility measures of candidates across different groups are actually comparable and that the per-group bias is known a-priori. We believe that the group skew unawareness is a necessary condition for the justification of post-processing algorithms in general and we therefore explicitly do not search for the optimal solution in terms of utility. We will go into more depth on this in Section 3.3.

Our definition of *ranked group fairness* reflects the legal principle of group under-representation in obtaining a benefit (Ellis & Watson, 2012; Lerner, 2003). We formulate a criterion by applying a statistical test on the proportion of protected candidates on every prefix of the ranking, which should be above a certain minimum. We also show that the verification of the ranked group fairness criterion can be implemented efficiently after pre-computing a verification data structure that we call *mTree*. This tree contains all possibilities to create a ranking that satisfies ranked group fairness and we provide an algorithm to build and maintain it. We also provide an algorithm AdjustSignificance for the mTree adjustment due to multiple dependent hypothesis testing, where we compute α_{corr} such that the type-I-error is less or equal to α for the *entire tree*.

Finally, we propose an algorithm, named FA*IR, for producing a top-k ranking that maintains high utility while satisfying ranked group fairness, as long as there are "enough" protected candidates from each group to achieve the desired minimum proportions. Note that if a group of protected candidates is too small to satisfy ranked group fairness, the ranking is necessarily bound to under-represent them. Nevertheless, we output a ranking that follows our utility principles. We also present extensive experiments to evaluate the performance of our approach compared to a group-unaware method (the so-called "color-blind" method) with respect to both, the expected utility of a ranking and the fairness degree measured, in terms of expected exposure.

Summarizing, the main contributions of this paper are:

(1) the principled definition of *ranked group fairness* for multiple protected groups, and the associated Fair Top-k Ranking problem;
(2) the FA*IR algorithm for multiple protected groups to produce a top-k ranking that maximizes utility while satisfying ranked group fairness.
(3) a mathematical framework to solve the problem of multiple *dependent* hypotheses testing: we will see how to adjust the given significance level α to a corrected α_{corr}, such that the *overall* probability for a type-1-error equals the given significance level α.
(4) the algorithms ComputeMTree and AdjustSignificance which translate ranked group fairness into a pre-computed data structure. This will enable efficient verification of ranked group fairness at testing time and efficient creation of fair rankings at creation time.

Our method can be used within an anti-discrimination framework such as *affirmative actions* (Sowell, 2005). We do not claim these are the only way of achieving fairness, but we provide *an algorithm grounded in statistical tests that enables the implementation of an affirmative action policy in the context of ranking*.

The rest of this paper is organized as follows. The next section presents a brief survey of related literature, while Section 3 introduces our ranked group fairness and utility criteria and a formal problem statement. Section 4 presents a data structure (mTree) that allows fast verification of the ranked group fairness criterion. Section 5 presents a procedure for statistical test significance adjustment, which is required due to multiple hypotheses testing. Section 6 describes the FA*IR algorithm for fair rankings, while Section 7 presents experimental results. Finally, Section 8 presents our conclusions and future work.

2. Related work

Algorithmic Fairness is a relatively new computer science topic, with most research published during the last ten years (Hajian et al., 2016; Rosenbaum & Fichman, 2019). Some proposals are oriented to discovering and measuring discrimination (*e.g.*, Angwin, Larson, Mattu, and Kirchner (2016), Bonchi, Hajian, Mishra, and Ramazzotti (2017), Pedreshi et al. (2008)); while others deal with mitigating or removing discrimination to ensure that the results of different algorithms do not lead to discriminatory decisions (*e.g.*, Dwork et al. (2012), Hajian and Domingo-Ferrer (2013), Hajian, Domingo-Ferrer, and Farràs (2014), Kamiran, Calders, and Pechenizkiy (2010), Zemel, Wu, Swersky, Pitassi, and Dwork (2013)). All these methods are known as *fairness-aware algorithms*.

2.1. Group fairness and individual fairness

Two basic frameworks have been adopted in recent studies on algorithmic discrimination: (i) *individual fairness*, a requirement that individuals should be treated consistently (Dwork et al., 2012; Žliobaitė, 2015); and (ii) *group fairness*, a requirement that the protected groups should be treated similarly to the advantaged group or the population as a whole (Pedreschi, Ruggieri, & Turini, 2009b; Pedreshi et al., 2008). Other work speaks of *inter-* and *intra-group fairness* (Beutel et al., 2019; Grgić-Hlača, Zafar, Gummadi, & Weller, 2017; Yeom & Tschantz, 2021), where inter-group fairness is concerned with fair assignment of outcomes for items from different groups, and intra-group fairness relates to items that belong to the same group.

Different fairness-aware algorithms have been proposed to achieve group and/or individual fairness, mostly for predictive tasks. Calders and Verwer (2010) consider three approaches to deal with naive Bayes models by modifying the learning algorithm. Kamiran et al. (2010) modify the entropy-based splitting criterion in decision tree induction to account for attributes denoting protected groups. Kamishima, Akaho, Asoh, and Sakuma (2012) apply a regularization (*i.e.*, a change in the objective minimization

function) to probabilistic discriminative models, such as logistic regression. Zafar, Valera, Rodriguez, and Gummadi (2015) describe fairness constraints for several classification methods. Feldman, Friedler, Moeller, Scheidegger, and Venkatasubramanian (2015) study *disparate impact* in data, which corresponds to an unintended form of group discrimination, in which a protected group is less likely to receive a benefit than a non-protected group (Barocas & Selbst, 2016). A technically similar framework has been proposed by Zehlike, Hacker, and Wiedemann (2020) for a setting with multiple protected groups, which continuously interpolates between group fairness as statistical parity and individual fairness. We use this method as one of our experimental baselines in Section 7.2. Their method uses optimal transport to calculate a fair score representation for each group, which is the Wasserstein-barycenter of all group distributions. It also enables a "continuous interpolation" that generates protected group scores between the original distribution in the protected groups and the distribution of the barycenter; this is controlled by a parameter but there are no guarantees on the proportions along the ranking. Recently, other fairness-aware algorithms have been proposed for mostly supervised learning algorithms and different bias mitigation strategies (Celis, Deshpande, Kathuria, & Vishnoi, 2016; Corbett-Davies, Pierson, Feller, Goel, & Huq, 2017; Friedler, Scheidegger, & Venkatasubramanian, 2016; Hardt, Price, & Srebro, 2016; Jabbari, Joseph, Kearns, Morgenstern, & Roth, 2016; McNair, 2018). Hardt et al. (2016) study fairness in terms of *equalized odds* and proposes methods to ensure equal precision across groups. Zafar, Valera, Gomez Rodriguez, and Gummadi (2017) introduces a new concept called *disparate mistreatment* and proposes a method that seeks to reduce differences in error rates. McNair (2018) introduces a set of statistical tests, two frequentist methods (Cochran–Mantel–Haenszel test and beta regression) and one Bayesian method (Bayesian Model Averaging), to assess the likelihood of fair decisions.

In contrast, our paper considers the more general setting by achieving fairness in ranking algorithms and it even can be reduced to other problems such as classification (see Problem 1). Our framework provides an evaluation criterion, ranked group fairness, and offers guarantees under this criterion, which can be directly controlled by a external parameter. Moreover, it takes into account both frameworks of group and individual fairness.

2.2. Fair ranking

Algorithmic fairness for rankings is concerned with a sufficient representation of different groups and consistent treatment of individuals across all ranking positions (Castillo, 2018). This leads to the motivation of producing rankings that do not systematically place items from protected groups at the lower positions of the result list.

Yang and Stoyanovich (2016) propose a statistical parity measure based on comparing the distributions of protected and non-protected candidates on different prefixes of the list and then averaging these differences in a discounted manner. We use the synthetic ranking generation procedure of Yang and Stoyanovich (2016) to calibrate our method, and optimize directly the utility of a ranking that has statistical properties (ranked group fairness) resembling the ones of a ranking generated using that procedure; in other words, unlike them, we connect the creation of the ranking with the metric used for assessing fairness. Moreover, we also take into account the individual fairness criteria.

Regarding how to evaluate fairness/bias in a ranking, Kulshrestha et al. (2017) propose a quantification framework that measures the bias of the results in a search engine. This framework discerns to what extent this output bias is due to the input dataset that feeds into the ranking system, and how much is due to the bias introduced by the system itself. Kuhlman, VanValkenburg, and Rundensteiner (2019) also propose a quantification and diagnostics framework using pairwise error metrics as means of evaluation for fairness in rankings. Another quantification and bias mitigation framework is presented by Geyik, Ambler, and Kenthapadi (2019), which can be tuned towards different definitions of fairness such as demographic parity and so-called equal opportunity. Yang et al. (2018) present a collection of metrics for fairness in ranked outputs which is inspired by nutrition labels.

As a general framework for constrained ranking, Celis, Straszak, and Vishnoi (2018) propose algorithms in which the constraint is a $k \times \ell$ matrix with k being the length of the ranking and ℓ the number of classes, indicating the maximum number of elements of each class (protected or non-protected in the binary case) that can appear at any given position in the ranking. Kuhlman and Rundensteiner (2020) propose algorithms for rank aggregation, in which we are given various rankings on the same items (e.g., rankings of the same candidates by each member of a committee) and we would like to produce a consensus ranking that guarantees fairness for disadvantaged groups of candidates.

Singh and Joachims (2018) introduce the concept of exposure or attention given to a group, incorporating the empirical observation that the probability for items to be examined by a searcher decreases quickly with each lower position. Parallel with Biega, Gummadi, and Weikum (2018), they propose an integer linear program that receives a vector of relevance scores and ranks items accordingly subject to constraints in disparate attention across groups. They also introduce a time component into their optimization problem, such that their notion of disparate attention is to be considered over time.

Lahoti, Gummadi, and Weikum (2019) focus on *individual fairness* for rankings and propose an advancement of its definition by Dwork et al. (2012) in such way that it no longer relies on a human specification of a distance metric.

Zehlike and Castillo (2020) introduce the first bias mitigation algorithm within a learning to rank framework, by defining the learning objective in terms of accuracy as well as in terms of exposure fairness of the result ranking. The learning objective is concerned with disparate impact, as in the work by Shang, Sun, and Lam (2020), whereas Beutel et al. (2019) and Singh and Joachims (2019) introduce fairness objectives in learning to rank methods that comply with disparate treatment considerations.

The referenced works cannot be directly compared with the method we propose in this paper. Some of them require the number of protected candidates at each position to be given as an input (Celis et al., 2018), while we determine the required number of candidates, or require input preferences to be given as rankings to be aggregated (Kuhlman & Rundensteiner, 2020). Others introduce metrics of exposure (Singh & Joachims, 2018), amortized exposure (Biega et al., 2018), or NDCG variants (Yang & Stoyanovich, 2016), or ratios ("skew") or KL-divergence based metrics (Geyik et al., 2019) that do not match the probability-based metric that we use. In our experiments, we compare with CFAθ (Zehlike, Hacker, et al., 2020), which is an utility-based framework such as ours, and one in which utilities can only be considered comparable within a group, but not across groups.

2.3. Diversity

The Information Retrieval community has studied for many years how to avoid showing only items of the same class in a ranking. A motivation to investigate this problem is that user queries may have different intents and the system should cover several of them with answers. A common approach to solve this, since Carbonell and Goldstein (1998), is to retrieve a "diverse" but relevant set of items. This is achieved through on one side, maximizing the overall relevance (utility) and on the other side, considering the distances between elements and adding a penalty for selecting an item that is too similar to an item already ranked at a higher position. Another commonly used definition is that diversity should be understood as a way to incorporate uncertainty about user intent, in the sense that all queries have some degree of ambiguity (Agrawal, Gollapudi, Halverson, & Ieong, 2009). Other works Channamsetty and Ekstrand (2017), Kunaver and Požrl (2017) are concerned with different types of bias such as presentation bias, where only a few items are shown and most of the items are not shown, as well as popularity bias and a negative bias towards new items. An exception is the work of Sakai and Song (2011), which provides a framework per-intent for evaluating diversity. In this framework, an "intent" can be mapped to a protected/non-protected group in the fairness ranking setting. However, their method is concerned with evaluating a ranking, similar to the NDCG-based measures described by Yang and Stoyanovich (2016) and that we described before, and not with the construction of such a ranking, as we do in this work.

In contrast to most of the research on diversity of ranking results or recommendation systems, our work operates on a set of discrete classes and not on similarity measures to previous items. Furthermore, *we are not only concerned with the utility that search- and recommendation system users receive*, but also with the exposure received by the ranked items, which can represent individuals, places or organizations. Another main difference is that diversification is usually symmetric yielding interchangeable groups, while *fairness-aware algorithms are mostly asymmetric*, because they focus on increasing the overall benefit received by a protected or disadvantaged group.

2.4. Fairness in recommender systems

Next line of complimentary related work is looking at algorithmic bias in recommender systems (Burke, Sonboli, & Ordonez-Gauger, 2018; Chakraborty, Patro, Ganguly, Gummadi, & Loiseau, 2019; Edizel, Bonchi, Hajian, Panisson, & Tassa, 2020; Ekstrand & Kluver, 2021; Jannach, Lerche, Kamehkhosh, & Jugovac, 2015; Kamishima, Akaho, Asoh, & Sakuma, 2018; Steck, 2018; Yao & Huang, 2017; Zhu, Hu, & Caverlee, 2018). Recommendation and personalization are useful technologies which influence more and more our daily decisions. Recent works show that biased recommendations can reinforce stereotypes such as those based on gender or ethnicity, possibly resulting in disparate impact. Even when the recommender system does not take as input any demographic information: nevertheless, by carefully exploiting items' and users' similarities, the algorithm might end up recommending an item to a very homogeneous set of users. For example, the algorithm would be considered biased if displays ads for high-income jobs to men much more often than it does to women. Thus, even when recommendations are of high quality, they might be not well perceived by the user that might find them "too accurate" and discriminatory. To address such biases in recommender systems, different mitigation approaches have been proposed such as limiting the predictability of sensitive features from the user–item recommendation matrix (Edizel et al., 2020; Kamishima et al., 2018).

3. The fair top-k ranking problem for multiple protected groups

In the following, we introduce the needed notation (Section 3.1), and present the definition of ranked group fairness (Section 3.2), as well as definitions of utility (Section 3.3). Lastly we present a formal problem statement (Section 3.4). Our formalization extends the one previously presented for a binomial setting (one protected group) (Zehlike et al., 2017) into a multinomial setting. We explicitly frame this as an extension rather than an entirely new framework, and hence the definitions in this paper follow those from (Zehlike et al., 2017). The notation used in this paper is summarized on Table 2.

3.1. Preliminaries and notation

For our setting we examine a set of candidates $[n] = \{1, 2, \dots, n\}$, where k of them will be selected based on their respective qualification. We denote this "utility" of candidate c for each $c \in [n]$ by q_c, which can be viewed as the fitness of candidate c for the task at hand, or *e.g.*, a search query. We assume this score to be given and denote that it may be a weighted combination of several different attributes. If this utility is computed by a machine learning model, an effort must be done to prevent preexisting and technical biases towards a protected group to be embodied in the model (see, *e.g.*, (Sweeney, 2013)). We define a set of candidate groups $G = \left\{g_0, g_1, \dots, g_{|G|}\right\}$ that form a partition of $[n]$. We will consider g_0 as non-protected (or privileged) and all remaining groups as protected (or disadvantaged). We assume there are at least k candidates in each group.

Let $\mathcal{P}_{k,n}$ denote all subsets of $[n]$ with exactly k elements and let $\mathcal{T}_{k,n}$ represent the union of all permutations of sets in $\mathcal{P}_{k,n}$. Thus $\mathcal{T}_{k,n}$ forms the solution space of the top-k ranking problem. For a permutation $\tau \in \mathcal{T}_{k,n}$ and an element $c \in [n]$, let

$$r(c, \tau) = \begin{cases} \text{rank of } c \text{ in } \tau & \text{if } c \in \tau\ , \\ k+1 & \text{otherwise.} \end{cases}$$

By τ_g we denote the number of elements of group g that are present in τ, i.e. $\tau_g = |\{c \in \tau \wedge c \in g\}|$. We set $\tau_G = \langle \tau_g \rangle_{g \in G}$, i.e., the vector that contains these numbers for each group.

Table 2
Notation.

Candidates	
$[n]$	Set of candidates
q_c	Qualifications of candidate c
$g_c \in G$	0 if candidate c is in the non-protected group, > 0 otherwise
$\|G\|$	The number of protected groups
Rankings	
$\mathcal{T}_{k,n}$	All permutations of k elements of $[n]$
τ	One such permutation
$r(c, \tau)$	The position of candidate c in τ, or $\|\tau\| + 1$ if $c \notin \tau$
$\tau_G = (\tau_1, \tau_2, \dots, \tau_{\|G\|})$	Vector of number of elements from group g in τ
cb	The "color-blind" ranking of $[n]$ by decreasing q_c
Group fairness criteria	
$p_G = (p_1, p_2, \dots, p_{\|G\|})$	Vector of minimum proportions for candidates of each protected group $g > 0$. The proportion p_0 of the non-protected group is implicitly given by $p_0 = 1 - \sum_{g \in G} p_g$.
α	Significance value for ranked group fairness test
α_{corr}	Adjusted significance for each fair representation test
Individual fairness criteria	
$\texttt{utility}(c, \tau)$	Qualification difference between candidate c and the least qualified candidate ranked above c while $q_c > q_d$
Model adjustment	
P_{fail}	Probability that our test fails on a fair ranking
$m_{\alpha,p}(k)$	Minimum number of protected candidates in top k positions; a vector of integers in case of $\|G\| > 1$

Let $cb \in \mathcal{T}_{n,n}$ be the total ranking of all candidates by decreasing utility: $\forall u, v \in [n], u = cb(i), v = cb(j), i < j \implies q_u \geq q_v$. If some utilities q_c are equal, we break ties arbitrarily. We refer to this as the *color-blind* ranking of candidates in $[n]$, because it simply focuses on their utility but ignores their protection status. Let $cb|_k = \langle cb(1), cb(2), \dots, cb(k) \rangle$ be a prefix of size k of this ranking.

Fair top-k ranking criteria. To solve the fair top-k ranking problem, our method searches for a particular ranking $\tau \in \mathcal{T}_{k,n}$ that maximizes the following objectives:

Criterion 1. Ranked group fairness: τ should fairly represent each protected group $g \in G$;
Criterion 2. Expected selection utility: τ should contain the most qualified candidates; and
Criterion 3. Expected ordering utility: τ should be ordered by decreasing qualifications.

The formal problem statement is given in Section 3.4, but first, we need to formally define each criterion, which we do in the next sections.

3.2. Group fairness for rankings

Criterion 1. is operationalized as a *ranked group fairness criterion* with two input parameters: (i) τ_G, the vector containing the number of candidates from each protected group in ranking τ, and (ii) p_G, a vector containing minimum target proportions for each protected group. Instinctively, a ranking is said to not fulfill the ranked group fairness condition, if the number τ_g of candidates from a protected group g falls far below the required number according to the given target proportions. Additionally, this criterion looks at the ordering in which those candidates appear within the ranking by comparing the actual number of protected elements from each group τ_G with their expected number. This number is calculated using the stochastic process of a multinomial distribution (i.e. the roll of a $|G|$-sided dice), and simulates that candidates were picked at random at each position if the dice shows the side of their group. This is a translation of the idea of luck-egalitarianism, which is a subcategory of substantive equality of opportunity (see, e.g., Heidari, Loi, Gummadi, and Krause (2019)). The process is repeated for *every prefix* of the ranking.

Definition 3.1 (*Multinomial Cumulative Distribution Function*)**.** Let $n \in \mathbb{N}$ be a number of trials where each trial results in one of the events $E_0, E_1, E_2, \dots, E_{|G|}$ and on each trial E_j occurs with probability p_j.

Let then X be a set of random variables that is multinomially distributed $X \sim \text{Mult}(n, p)$ with parameters n and $p = \langle p_0, p_1, p_2, \dots, p_{|G|} \rangle$, and let X_j be the number of trials in which event E_j occurs.

We then define $F(X; n, p) = P\left(E_1 \leq X_1, E_2 \leq X_2, \dots, E_{|G|} \leq X_{|G|}\right)$ the multinomial cumulative distribution function which computes the probability that each event E_j occurs at most X_j times in n trials given probabilities p.

With the multinomial CDF (The Pennsylvania State University, 2018) the ranked group fairness criterion is formalized as a statistical significance test. To express the probability of rejecting a fair ranking (i.e. a type-I-error) we include a significance parameter ($\alpha \in [0, 1]$).

Definition 3.2 (*Fair Representation Condition*)**.** Let $F(X; n, p)$ be the multinomial cumulative distribution function as defined above.

A ranking $\tau \subseteq \mathcal{T}_{k,n}$, having τ_G protected candidates from each group fairly represents all protected groups with minimal proportions $p_G = \langle p_1, \ldots, p_{|G|} \rangle$ and significance α,

if $F(\tau_G; k, p_G) > \alpha$.

The minimum proportion p_0 of the non-protected group is implicitly given by $p_0 = 1 - \sum_{g \in G} p_g$ and we will not include it explicitly in p_G to be consistent with τ_G.

The fair representation condition implements a statistical test, with null hypothesis H_0 assuming that all protected groups are represented with a minimal proportion. The alternative hypothesis H_a translates into an insufficient proportion of protected elements. Our test setup sets the p-value to $F(\tau_G; k, p_G)$ and we reject H_0, hence announce the ranking under test to be unfair, if the p-value is less than or equal to the threshold α. Note that according to this definition, in the case of a set of size one, either the element is in the protected group, and then we satisfy fair representation, or the element is not in the protected group, and then we satisfy fair representation if $1 - F > \alpha$.

To obtain the ranked group fairness condition, we require the fair representation condition to be met for all prefixes of the ranking:

Definition 3.3 (*Ranked Group Fairness Condition*)**.** A ranking $\tau \in \mathcal{T}_{k,n}$ satisfies the ranked group fairness condition with parameters p_G and α, if for every prefix $\tau|_i = \langle \tau(1), \tau(2), \ldots, \tau(i) \rangle$ with $1 \leq i \leq k$, the set $\tau|_i$ satisfies the fair representation condition with group target proportions p and significance $\alpha_{corr} = \textsc{AdjustSignificance}(k, p_G, \alpha)$.

Function $\textsc{AdjustSignificance}(k, p_G, \alpha)$ is a corrected significance to account for multiple hypotheses testing (described in Section 5).

We note there exists a solution space of rankings that satisfy this condition for a given 3-tuple (k, p_G, α), instead of just a single ranking as it is the case when only one protected group is present. We further note that a lower significance α translates into a lower probability of wrongly declaring a fair ranking as unfair, and we therefore use a relatively conservative setting of $\alpha = 0.1$ in our experiments (Section 7). It is possible to convert the binary choice (fair vs. unfair) of the ranked group fairness condition into a *ranked group fairness metric*. This metric would be the maximum $\alpha \in [0, 1]$ for which a ranking τ satisfies the ranked group fairness condition, given minimum proportions p_G. In this metric space, larger values imply a stronger compliance with the required number of protected elements from the top-positions onward.

3.3. Utility

Notions of utility aim to measure desiderata of rankings like ranking well qualified candidates as high as possible. Previous works (Celis et al., 2018; Yang & Stoyanovich, 2016) assumed to know the exact utility contribution of a candidate at a specific position. In contrast to that, we base our utility calculation on losses due to non-monotonicity (i.e., due to candidates not being ordered by decreasing scores anymore to satisfy the ranked group fairness constraint). This can also be understood as a measure of individual unfairness, as it calculates the largest score difference between a high-scoring candidate ranked below a low-scoring candidate.

Ranked utility. The ranked individual utility loss of a candidate c in a ranking τ is high, if another candidate with a low score was ranked at a better position.

Definition 3.4 (*Ranked Utility of an Element*)**.** The ranked utility of an element $c \in [n]$ in ranking τ, is:

$$\texttt{utility}(c, \tau) = \begin{cases} \overline{q} - q_c & \text{if } \overline{q} < q_c \text{where } \overline{q} \triangleq \min_{d : r(d,\tau) < r(c,\tau)} q_d \\ 0 & \text{otherwise} \end{cases}$$

Thus, a candidate c achieves the maximum ranked individual utility of zero, if there was no candidate with a lower score ranked at a better position in the ranking.

Selection utility. In order to satisfy Criterion 2. we define *selection utility* of a ranking. We will use this objective to prefer rankings in which more qualified candidates are included and less qualified are excluded.

Definition 3.5 (*Selection Utility*)**.** The selection utility of a ranking $\tau \in \mathcal{T}_{k,n}$ is

$$\min_{c \in [n], c \notin \tau} \texttt{utility}(c, \tau).$$

Analogous to the ranked utility of an element, a "color-blind" ranking of k items $cb|_k$ has a maximum selection utility of zero, if all candidates included in the ranking have higher scores than the candidates excluded from the ranking.

Ordering utility and in-group monotonicity. In order to satisfy Criterion 3. of we introduce *ordering utility* and an *in-group monotonicity constraint* as objectives. Both are necessary in order to prefer rankings in which candidates with higher scores are ranked above candidates with lower scores.

Definition 3.6 (*Ordering Utility*)**.** The ordering utility of a ranking $\tau \in \mathcal{T}_{k,n}$ is

$$\min_{c \in \tau} \texttt{utility}(c, \tau).$$

A rankings' ordering utility is only concerned with the worst ranked individual utility a candidate attains in it. In contrast to that, we define the in-group monotonicity constraint with respect to all elements in the ranking. The constraint guarantees, that, within groups, all candidates must be sorted by their scores.

Definition 3.7 (*In-Group Monotonicity*). A ranking $\tau \in \mathcal{T}_{k,n}$ satisfies the in-group monotonicity condition if $\forall c, d$ s.t. $g_c = g_d$, $r(c,\tau) < r(d,\tau) \Rightarrow q_c \geq q_d$.

A "color-blind" ranking of length k, $cb|_k$ has a maximum ordering utility of zero and it also satisfies the in-group monotonicity constraint.

Connection to the individual fairness notion. In contrast to taking distributive measures like the average utility, our notion of utility focuses on individuals by using the "worst-off" candidates. While other choices are possible, this has the advantage that we can only maximize utility through improving the outcome of specific individuals. Those people are the "worst-off" because they are excluded from a ranking in which others have lower scores or because they a ranked below other candidates with lower scores. Our definitions connect to notions of individual fairness, which require consistent treatment of individuals (Dwork et al., 2012). Two candidates with equal scores or qualifications should be treated equally. In our framework, a deviation from this equal treatment will end in a loss of utility. Thus, any trade-off can be measured explicitly and traced back to an individual.

3.4. Formal problem statement

The criteria we have described enable two different problem statements. First, we can use ranked group fairness as a constraint and maximize ranked utility. Second we can use ranked utility as a constraint and then maximize ranked group fairness.

Problem 1 (*Fair top-k ranking*). Given a set of candidates $[n]$, a partition of $[n]$ in groups $G = \{g_0, g_1, \ldots, g_{|G|}\}$, the vector p_G of minimum proportions per group, and parameters $k \in \mathbb{N}^+$ and $\alpha \in [0,1]$, produce a ranking $\tau \in \mathcal{T}_{k,n}$ that:

(i) satisfies the in-group monotonicity constraint;
(ii) satisfies ranked group fairness with parameters p_G and α;
(iii) achieves high selection utility subject to (i) and (ii); and
(iv) achieves high ordering utility subject to (i), (ii), and (iii).

Related problems. Other problem definitions with respect to the general criteria described in are possible, however one has to carefully consider their implications on a probabilistic fairness assumption. For instance, instead of maintaining high selection and ordering utility, we may seek to always find the one ranking τ^* from all possible fair rankings that *maximizes* selection and ordering utility. We acknowledge that this seems to be a tempting "optimization" of FA*IR but we believe that such a strategy is not fully compliant with multinomial ranked group fairness anymore (see Section 6.4 for more details).

4. Ranked group fairness verification

To verify ranked group fairness efficiently in time $O(k)$, a pre-computed data structure can be used, which is obtained by the *inverse multinomial CDF* with parameters k, p_G and α. The inverse CDF takes as input a probability (in our case multiple probabilities p_G) and returns that value of a random variable, at which the probability of said random variable being less than or equal to the returned value equals the input probability. As the multinomial CDF is not injective and hence has no inverse, there is no quantile function that tells us exactly how many candidates are needed at each k. Instead, there are various manifestations of τ_G that satisfy the fair representation condition $F(\tau_G; k, p_G) > \alpha$, which is why the verification data structure has the shape of a tree for each p_G, k and α.

Fig. 1 shows an example of such tree with $p_G = \langle 1/3, 1/3 \rangle$. Each tree level corresponds to the kth position in the ranking and are to be read from left to right, *i.e.*, the root level corresponds to the first ranking position and so forth. As an example, consider tree level 4 in Fig. 1. At this level, we have three nodes "(4, [2,0])", "(4, [1,1])", and "(4, [0,2])" each of them containing a set of minima for elements of the protected groups (the nodes do not include any minima for the non-protected group). This means it is acceptable to have among the first four elements in the ranking either at least 2 elements from protected group 1, or at least 1 element from each protected group, or at least 2 elements from protected group 2. Note however, that nodes have parental relationships and that each *path* corresponds to a fair distribution of protected candidates in the ranking. Thus, if at level 4 we rank two candidates from protected group 1, thus satisfying the node with the asterisk in Fig. 1, at level 5 have to satisfy either node "(5, [2,1])" or node "(5, [3,0])". The other nodes "(5, [1,1])", "(5, [1,2])", and "(5, [0,3])" are not a child of node "(4, [2,0])" and therefore cannot be considered to satisfy the ranked group fairness condition anymore. The tree is symmetric when the minimum proportions $p_g \in p_G$ are equal, and asymmetric when they are different. We note that it is possible to trade-off under-representation in one protected group against over-representation in another protected group, but only up to a point: the representation of every protected group cannot fall below a certain minimum. Additionally, when using this method to construct a ranking, we choose the most likely path, not any path, and thus avoid extremely unbalanced situations.

As stated at the beginning of this section mTrees are pre-computed data structures that allow efficient verification of the ranked group fairness condition. To construct them we use Algorithms 1 and 2. The first algorithm ComputeMTree takes a triple (k, p_G, α_{corr})

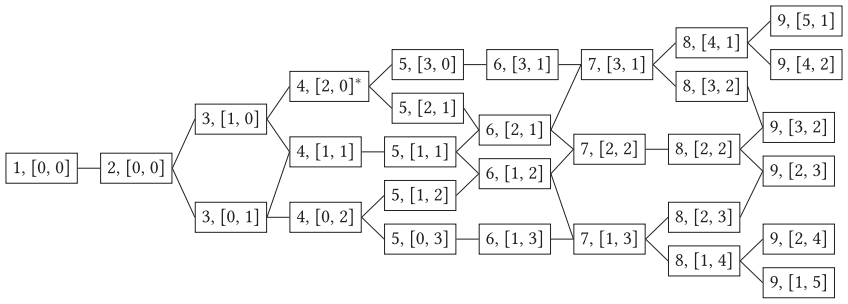


Fig. 1. Example of an mTree with two protected groups with minimum proportions $p_G = \langle 1/3, 1/3 \rangle$ and $\alpha = 0.1$. The notation $(k, [x, y])$ indicates that at ranking position k, we need at least x elements of group 1 and y elements of group 2 to satisfy ranked group fairness; group 0, which is the non-protected group, is always unconstrained. For instance, the node marked "(4, [2, 0])*" indicates that in the first 4 positions of this ranking, one of the acceptable configurations is to have 2 or more elements from group 1 and 0 or more elements from group 2. We see that in case of multiple protected groups, there are various ways of satisfying Definition 3.3. Thus, each path in the tree corresponds to one valid strategy to place protected candidates in the ranking.

Algorithm 1: Algorithm ComputeMTree computes the data structure to efficiently verify or construct a ranking that satisfies multinomial ranked group fairness.

input : k, the size of the ranking to produce; p_G, the expected proportions of protected elements for each group; α_{corr}, the significance for each individual test.
output : mtree: A tree data structure that contains the minimum number of protected candidates for each group.

```
mtree[0] ← zeros(|G|) // initialize auxiliary root node with |G| entries
for i ← 1 to k do
    for parent in mtree[i − 1] do
        // find all child nodes that satisfy ranked group fairness
        children ← InverseMultinomialCDF(α_corr; i, p_G, parent)
        parent.children ← children
    end
end
return mtree
```

Algorithm 2: Algorithm InverseMultinomialCDF computes the inverse of the multinomial cumulative distribution function $F^{-1}(\alpha_{corr}; i, p_G)$. It finds all possible child nodes of a given parent that satisfy the ranked group fairness condition.

input : `parent`, the node of which we calculate all minimum target children;
i, the current position in the ranking;
p_G, the vector of expected proportions of protected elements of each group;
α_{corr}, the significance for each individual test.
output : `children`: A list of nodes with minimum targets that satisfy ranked group fairness

```
children ← {}
child ← copy(parent)
mcdf ← F(child; i, p_G)
if mcdf > α_corr then
    // if the multinomial cdf is greater than α_corr we do not need to increase the number of required protected candidates
    children.add(child)
end
else
    for j ← 1 to |G| do
        // test whether the multinomial cdf is greater than α_corr, if one more candidate of group j was required at postion i
        temp ← child
        temp[j] ← temp[j] + 1
        mcdfTemp ← F(temp; i, p_G)
        if mcdfTemp > α_corr then
            // if yes, append the new requirement to the mTree
            children.add(temp)
        end
    end
end
return children
```

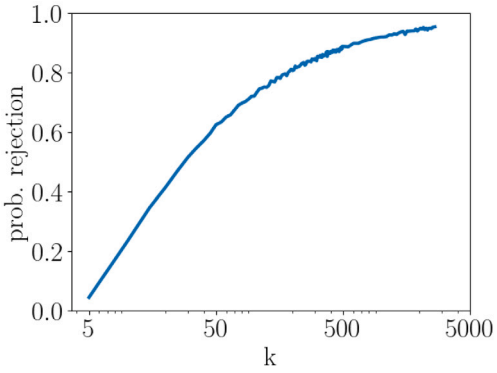


Fig. 2. Experiments on data generated by a simulation, showing the need for multiple tests correction. The data has two protected groups, rankings of lengths k are created by the stochastic process of a multinomial distribution ("rolling a 3-sided dice") with $p_G = (0.33, 0.33)$. These rankings should have been rejected as unfair at a rate $\alpha = 0.1$. However, we see that the rejection probability increases with k. Note the scale of k is logarithmic.

as input and returns the mTree for these parameters. First it creates an auxiliary root node that contains $|G|$ zero entries, which serves as the root parent. Then, for each parent node `parent` at each position $i \leq k$ it calls the inverse multinomial CDF function. The second algorithm InverseMultinomialCDF takes as input a 4-tuple $(\alpha_{corr}, i, p_G, \texttt{parent})$ and returns all nodes satisfying the following two conditions: (i) they have to be children of node `parent`, and (ii) they satisfy the fair representation condition (Definition 3.2).

5. Model adjustment

Recall that we restrict the probability of a type-I-error in our ranked group fairness test to α (Definition 3.2). To verify ranked group fairness, we test if a ranking satisfies the *fair representation condition* (Definition 3.2) for each prefix of size $1, 2, \ldots, k$. Because of this multiple hypotheses testing, if we perform each test with significance $\alpha_{corr} = \alpha$, we risk rejecting fair rankings at a rate larger than α. Hence, we require an adjusted significance $\alpha_{corr} = \text{AdjustSignificance}(\alpha, k, p)$ for each fair representation test.

Fig. 2 illustrates what happens, if we do correct the significance level α. In the figure, we assume there are two protected groups with $p_1 = p_2 = \frac{1}{3}$, and we generate inherently fair rankings of various lengths from $k = 5$ to $k = 5,000$. We test each ranking for fair representation with $\alpha_{corr} = \alpha$ and plot the probability of a type-I-error (declaring this fair ranking as unfair). We observe that the probability of rejecting a fair ranking as unfair increases with the number of hypotheses tests, i.e. with k.

If each of the k tests performed on a ranking were independent, we could use Šidák's correction, and set $\alpha_{corr} = 1 - (1 - \alpha)^{1/k}$. However, the tests are not independent, as the tests with prefixes k_1, k_2 (for instance), overlap on their first $\min(k_1, k_2)$ elements. In this section, we present a procedure to adjust the significance level under dependent hypotheses tests, such that an mTree has an overall probability for a type-I-error of α.

5.1. General procedure

With the presence of more than one protected group, the analytical extension of the model adjustment to a multinomial setting is too complex to be written into a closed formula. In contrast to the model adjustment for one protected group which we used in (Zehlike et al., 2017), we found no analytical way to calculate all permutations which pass or fail the test. Therefore, we develop an empirical procedure to adjust α:

(1) Get input p_G, k, α.
(2) Build mTree with input $\alpha_{corr} = \alpha$.
(3) Create M rankings by rolling a biased $|G|$-sided dice with each side's probability to show corresponding to a minimum proportion in vector p_G.
(4) Test all those rankings against the mTree and count how many tests fail.
Remember that we want to observe a maximum failure probability of $P_{\text{fail}} = \alpha$, because all rankings created by this multinomial stochastic process are considered to be inherently fair.
(5) If $P_{\text{fail}} > \alpha$, we choose a new α_{corr} using a binary search heuristic.
(6) Now we build a new mTree using α_{corr} and repeat the procedure until $P_{\text{fail}} \approx \alpha$.

Once we found α_{corr} we can recompute the mTree from Fig. 1 to obtain an overall significance level of $\alpha = 0.1$. Fig. 3 shows the adjusted mTree with the same parameters p_G.

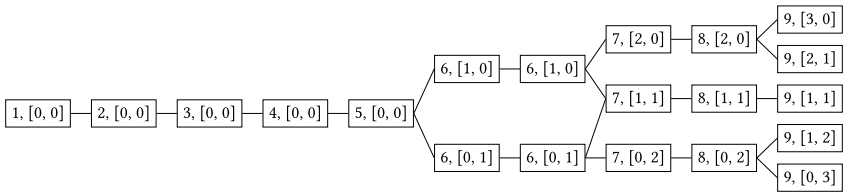


Fig. 3. Example of an mTree with two protected groups with minimum proportions $p_G = [1/3, 1/3]$ and $\alpha_{corr} = 0.1$. Compared to Fig. 1 this tree is less strict such that its *total* probability α_{corr} of rejecting a fair ranking (i.e. a type-1-error) is 0.1.

5.2. Optimizations for the MTree calculation

In this subsection we explain how we optimize the adjustment procedure from the previous section to reduce complexity. At each iteration, it requires to compute a new mTree, which is expensive on its own. Furthermore, depending on the level of accuracy needed, a large number of iterations M might be required and the binary search heuristic may need many steps to find α, if large intervals have to be searched. We use three strategies to drastically reduce computational costs of this simulation-based adjustment: (i) we reduce the space requirements of the mTree structure; (ii) we exploit the monotonicity of α_{corr} with respect to α and use a regression procedure to speed up the binary search. (iii) we apply the adjustment procedure for small trees first and only increase k, if we found the correct α_{corr} for the small trees.

5.2.1. Reducing mTree space requirements

We improve the mTree data structure by excluding the calculation and storage of redundant information. First, we store each node only once at each level and duplicate nodes are combined into a single node with multiple parents.

Second, in case of equal minimum proportions for all groups $p_1 = p_2 = \cdots = p_{|G|}$ the mTree shows a convenient property that we can use to reduce additional space, as well as computation time. Remember that whenever the multinomial CDF value falls below α for a particular position i, we have to put a protected candidate onto i. For equal minimum proportions the tree branches into $|G| - 1$ symmetric nodes $m(i)$ of the same likelihood. As an example reconsider the mTree from Fig. 3 at level 6. For two protected groups with minimum proportions $[1/3, 1/3]$ we see that the tree branches into two symmetric nodes $(6, [1, 0])$ and $(6, [0, 1])$. Both have the same multinomial CDF values. We store only one of the nodes and flag it as "has mirrored node" and continue our mTree computation only in the stored branch. This way we save half of the space and computation time needed, without losing any information about the tree.

Furthermore, we reduce the size of the mTree by actually leaving out the parent–child relationship and merely storing the nodes itself together with their respective depth levels. Fig. 3 shows the mTree structure with all parent–child relations as edges. We leave out these edges, thus reducing space, because we can prove that, if a single node exists on each level, which accepts a given ranking as fair, then a valid path to that node exists in the tree.

Before we can prove this property, we need to introduce the following definition, which formalizes what a successful mTree test at level i looks like.

Definition 5.1 (*Successful mTree Testing*)**.** Let τ be a ranking of size k and $\tau_{G,i} = (\tau_1, \ldots, \tau_{|G|})$ the numbers of ranked protected elements from group $1, \ldots, |G|$ up to position i. Let us assume MT be a *mTree*, with node $MT_{G,i} = [m_1(i), \ldots, m_{|G|}(i)]$ giving the number of protected candidates of group $1, \ldots, |G|$ required up to position i. We write $\tau_{G,i} \geq m_{G,i}$ if $\tau_g \geq m_g$ for all $g = 1, \ldots, |G|$. We call a test on level i of MT successful, iff $\tau_{G,i} \geq m_{G,i}$.

Again, in order to remove the parental relationships in the mTree, thus reducing storage space, we have to prove that, if we test a ranking on each level of the mTree successfully, the entire ranking will be fair according to the ranked group fairness definition. We prove this by showing that, if a ranking passes the test for any two nodes n_1 and n_2 at two consecutive levels h and $h + 1$, and n_1 is *not* a parent of n_2, then all actual children of n_1 will have a weaker requirement than n_2 and will hence also test successfully. Moreover, we show that all nodes at level $h + 1$ for which the ranking fails the test are part of a path that already rejected it as unfair at level h. Consider an example from Fig. 3 : Let us assume a ranking passes the test at level 8 with exactly the required protected items $(8, [1, 1])$. Now let us assume that at level 9, the given ranking would pass the test for node $(9, [2, 1])$, which is not a successor of $(9, [1, 1])$. In fact we see that the actual successor of $(8, [1, 1])$ is a node with the same requirements $(9, [1, 1])$. However, if our ranking passes the test for the stricter node $(9, [2, 1])$, it also passes for $(9, [1, 1])$ and thus we do not need to know the true parent of $(9, [2, 1])$. Note that the ranking would fail at node $(9, [3, 0])$, but with $\tau_G = (1, 1)$ at level 8 it would have failed already at $(9, [3, 0])$'s predecessor $(8, [2, 0])$.

Theorem 5.2. *Let MT be a mTree and τ a ranking of size k. There exists at least one successful test for τ at each level of MT, iff there exists a path from the root of MT to a leaf of MT and τ passes the test at each node on that path.*

Proof.

"⇐".: If there exists a path for which τ passes the test at each node, there also exists a node on each level of MT where τ passes the test because every path has a node on every level.

"⇒".: Let MT be a mTree and τ be a ranking that passes the test at level h of MT.

Let $m_{G,h} = [m_1(h), \dots, m_{|G|}(h)]$ be the node on level h that successfully tested τ.

Let further be $\tau_g(h)$ the number of protected candidates of group g ranked at up to position h.

Let $\sum_{g=1}^{|G|} |m_g(h) - \tau_g(h)| = 0$, meaning that the ranking includes the exact amount of required protected candidates at level h and not more.

Note that this assumption reduces the number of nodes at which τ passes the test at level h to a minimum.

If we assumed that τ included more protected candidates than required, we would only increase the number of nodes at level h for which τ would pass the test.

Thus, we can proof this direction with the above assumption without loss of generality.

Let $m_{G,(h+1)}$ be a node which tests τ successfully on level $h+1$ with $\sum_{g=1}^{|G|} |m_g(h+1) - \tau_g(h+1)| = 0$.

For all entries $m_g(h+1)$ of $m_{G,(h+1)}$ it is that $m_g(h) \leq m_g(h+1)$ by construction of the mTree, in which requirements for protected candidates can only increase or stay the same, but cannot decrease.

We can now distinguish between the following two cases:

Case 1: $m_{G,(h+1)}$ is a child of $m_{G,h}$.

Then they form a path.

Case 2: $m_{G,(h+1)}$ is not a child of $m_{G,h}$.

Let now $m'_{G,(h+1)}$ be a child of $m_{G,(h)}$.

Because of $\sum_{g=1}^{|G|} |m_g(h) - \tau_g(h)| = 0$, and a successful test at level h+1, the following inequations hold: $\sum_{g=1}^{|G|} |m_g(h) - m_g(h+1)| \leq 1$ and $\sum_{g=1}^{|G|} |m_g(h) - m'_g(h+1)| \leq 1$.

If $\sum_{g=1}^{|G|} |m_g(h) - m_g(h+1)| = \sum_{g=1}^{|G|} |m_g(h) - m'_g(h+1)| = 0$ or $\sum_{g=1}^{|G|} |m_g(h) - m_g(h+1)| = \sum_{g=1}^{|G|} |m_g(h) - m'_g(h+1)| = 1$, it follows that $m_{G,(h+1)} = m'_{G,(h+1)}$ and we would have a contradiction with the fact that $m'_{G,(h+1)}$ is not a child of $m_{G,h}$, because then the nodes would be equal or different by one unit in one position.

Hence, we need that either (2.1) $\sum_{g=1}^{|G|} |m_g(h) - m_g(h+1)| = 1$ and $\sum_{g=1}^{|G|} |m_g(h) - m'_g(h+1)| = 0$, or (2.2) $\sum_{g=1}^{|G|} |m_g(h) - m_g(h+1)| = 0$ and $\sum_{g=1}^{|G|} |m_g(h) - m'_g(h+1)| = 1$.

Case (2.1) means because of $\sum_{g=1}^{|G|} |m_g(h) - m_g(h+1)| = 1 > \sum_{g=1}^{|G|} |m_g(h) - m'_g(h+1)| = 0$ that the mTree accepts a ranking with one more protected candidate than required by the actual child of $m_{G,h}$, which contradicts our hypothesis that the ranking included the exact amount of required protected candidates. It follows that if we test $\tau_g(h+1)$ successfully with $m_{G,h+1}$ it would also satisfy $m'_{G,h+1}$. Case (2.2) is impossible according to algorithm 2. In detail, if $\sum_{g=1}^{|G|} |m_g(h) - m_g(h+1)| = 0$ it means that $m_g(h+1) = m_g(h)$ and therefore, $F(m_g(h); h+1, p_G) > \alpha$ (line 4 of algorithm 2). But if for the child of $m_{G,h}$, namely $m'_{G,h+1}$ it holds that $\sum_{g=1}^{|G|} |m_g(h) - m'_g(h+1)| = 1$, it means that $F(m_g(h); h+1, p_G) \leq \alpha$ so that we would have added a new node as a child of $m_g(h)$ according to lines $8-14$ of algorithm 2. Since both conditions cannot be true at the same time, Case (2.2) cannot occur.

In summary, Case (2.1) will only occur if we ranked more protected candidates than needed, and that the resulting test would be stricter than following a path through the mTree. We showed that Case (2.2) is impossible. There is only Case 1 left if we have ranked exactly the number of protected candidates needed at each level of the mTree. It follows that for any ranking that is tested successfully on each level of the mTree, it either was tested by nodes of a path through the mTree or was tested by a series of nodes which is stricter than such path. □

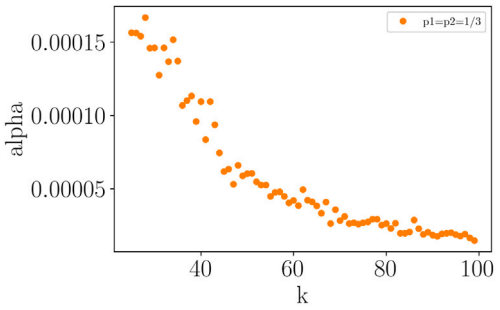
Because of Theorem 5.2 we do not need to keep the tree structure (i.e. parental relationship between nodes) and may store only a set of nodes for each level, while removing duplicate entries.

5.2.2. *Finding a good starting candidate for the binary search*

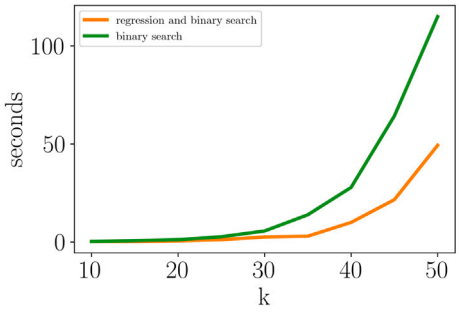
We use a second-degree polynomial regression model to get our first estimate for a good α_{corr} candidate and apply the binary search heuristic from that candidate, rather than starting with a random value, that might be very far away from the correct α_{corr}. We need a few additional parameters as input: `kTarget` — the length of the target ranking, `kStart` — the size of the first mTree, `maxPreAdjustK` — the maximum size of the mTree before we use regression to predict a good candidate α_{c_r} for the final α_{corr}, and `num_iterations` — the number of training instances to be computed. To create a training dataset R, we compute `num_iterations` small mTrees (i.e. with different $k \leq$ `maxPreAdjustK`) and adjust the respective α values as described in Section 5.1. For each iteration j the pair (k_j, α_{c_j}) is stored as a training instance in R. Fig. 3(a) shows a training set for $p_G = [1/3, 1/3]$, `maxPreAdjustK` $= 100$. Then a regression model is trained to predict α_{c_r} for `kTarget`. This α_{c_r} is now used to start the binary search for the correct and final α_{corr}. Fig. 3(b) shows the runtime difference for the model adjustment routine with and without the use of regression.

5.2.3. *Adjusting for small k first*

We use the fact that given a value of α, the mTree calculation is not dependent on k, i.e., a mTree for $k = 20$ and a mTree for $k = 10$ for the same α are equal in the first ten positions. This means that we can start the adjustment from the root node and expand the tree gradually to find the correct α_{corr}, because if P_{fail} is too high for given k, it will also be too high for any $k' > k$. We can see in Fig. 2 that P_{fail} grows very fast for small k, which makes an early adjustment of α most efficient to save computation time.



(a) Training data R for a regression model to predict a good candidate for α_{corr}. Each pair (k_j, α_{c_j}) is computed for small k using the procedure described in Subsection 5.1.



(b) Computation time comparison between binary search only and binary search combined with regression for the multinomial significance adjustment.

Fig. 4.

5.3. Final adjustment algorithm

Algorithm 3 shows the overall adjustment algorithm in pseudo-code, which performs the following steps:

(1) Define the necessary parameters: `kTarget`, `kStart`, `maxPreAdjustK`, and `num_iterations`
(2) Adjust α for a mTree of size `kStart` to get α_{c_1} using binary search.
(3) Add pair $\left(\texttt{kStart}, \alpha_{c_1}\right)$ to a regression training set R.
(4) Increase `kStart` by `stepsize` $= \frac{\texttt{maxPreAdjustK}}{\texttt{num\_iterations}}$
(5) Compute a mTree with parameters `kTarget`, p_G, α_{c_1} and adjust α_{c_1}.
(6) Name result α_{c_2} and add pair $\left(\texttt{kStart}, \alpha_{c_2}\right)$ to R.
(7) Repeat steps (4)–(6) until `kStart` == `maxPreAdjustK`.
(8) Train a regression model with training data R to predict α_{c_r} for `kTarget`.
(9) Use binary search (Algorithm 5) to find α_{corr} for parameters k, p_G, α_{c_r}.

5.4. Further optimizations

In addition to the optimizations *during* mTree calculation which we presented in Section 5.2, our implementation caches already computed mTrees and their components to never do the same computation twice. We remark that a mTree has to be computed only once for a particular combination of k, p_G, α.

5.4.1. MCDF cache

Table 3 shows that the highest computational cost arises from computing the multinomial cumulative distribution function F. In the worst case Algorithm 2 computes it $|G| + 1$ times for each group g in $m_g(i)$ and each position $i \leq k$. However, the same calculation may be done many times: As an example consider the (fictive) mTree nodes $(3, [2, 1])$ and $(3, [1, 2])$ at position $k = 3$. To compute the successors of node $(3, [2, 1])$ we call Algorithm 2 with arguments $(4, [2, 1])$, $(4, [3, 1])$ and $(4, [2, 2])$. We store the results of these calculation in a map that we call MCDF cache with the algorithm arguments (k and the minimum protected candidates of each group) as key and the corresponding mcdf as value. Next we compute the successors of node $(3, [1, 2])$ and call Algorithm 2 with arguments $(4, [1, 2])$, $(4, [2, 2])$ and $(4, [1, 3])$. We see that we would compute the MCDF for $(4, [2, 2])$ twice, but instead we can now read it from the MCDF cache. Note that the MCDF computation is only depends on p_G and not on $alpha$. We can therefore persist the MCDF cache on disk for a particular vector p_G and load it for any mTree calculation with the same p_G in the future.

5.4.2. Stored mTrees

During the computation of an adjusted mTree with parameters k, p_G, α we calculate many temporary mTrees (first the unadjusted ones, then the ones for the regression algorithm, then the ones for the binary search steps). We persist all of the temporary mTrees plus the final tree in files for later usage. The filenames contain the tree parameters, whether or not it is adjusted, and its probability to fail a fair ranking P_{fail}. If any of these trees is needed at a later point in time it can be loaded from disc instead of being recomputed, be it as input for multinomial FA*IR or as temporary tree during a new adjusted mTree computation.

5.5. Complexity analysis

This section presents time and space complexity analyses for all algorithms from this section. Table 3 shows the asymptotic costs for each algorithm without any pre-computed data (i.e. no mTree exists and the MCDF cache is empty). The complexity analysis for FA*IR is done in Section 6.2, but Table 3 already contains the summary, such that it shows time and space complexity for all algorithms in this paper.

5.5.1. InverseMultinomialCDF *complexity*

The complexity of the multinomial CDF is dependent on the current position i (also understood as *number of trials*), $|G|$ the number of protected groups, and $m_{G,i}$ the vector of minimum required protected candidates for each group at position i (also *number of successes*). We will write $\mathcal{O}(\text{MCDF}(i, p_G, \alpha))$ as the asymptotic complexity of this function. In the library we used for our implementation, the time complexity to calculate the multinomial CDF at position i is in $\mathcal{O}(\text{MCDF}(i, p_G, \alpha)) = \mathcal{O}(i^{|G|})$. Thus, the most expensive position to calculate the multinomial CDF for is k. For the overall time complexity of InverseMultinomialCDF we get $\mathcal{O}(|G|) \cdot \mathcal{O}(\text{MCDF}(k, p, \alpha))$ with $|G|$ being the number of protected groups. The space complexity is $\mathcal{O}(|G|^2)$ because we store $|G|$ nodes, with each node being an array of length $|G|$.

5.5.2. ComputeMTree *complexity*

Algorithm ComputeMTree constructs an mTree of depth k. There may be up to $|G|$ possible children for each node in the tree resulting in $|G|^{k-1} + 1$ nodes, leading to a time complexity of $\mathcal{O}(|G|^k) \cdot \mathcal{O}(\text{InverseMultinomialCDF})$. We store $|G|^{k-1} + 1$ nodes with arrays of length $|G|$, leading to space complexity of $\mathcal{O}(|G|^k)$.

5.5.3. MultinomialAlphaAdjustment *complexity*

Also, in the multinomial case we adjust α to α_c with a binary search heuristic. However there is no discrete measure for mTrees as was in case of the mTable mass. Therefore, the complexity of the binary search depends on the tolerance ϵ set beforehand as a stopping criteria. As we are searching on the interval $[0, \alpha[$ the resulting number of possible mTrees is $\frac{\alpha}{\epsilon}$. Thus the binary search needs $\mathcal{O}(\log \frac{\alpha}{\epsilon})$ time because we calculate one mTree and its failure probability for each step. The failure probability is computed experimentally by creating 10.000 rankings and testing them against the current mTree. This process needs $\mathcal{O}(10.000 \cdot k)$ to create the rankings plus $\mathcal{O}(k)$ to test each of them. Hence the overall time complexity becomes $\mathcal{O}(\log \frac{\alpha}{\epsilon}) \cdot (\mathcal{O}(k^2) + \mathcal{O}(\text{ComputeMTree}))$. The space complexity is in $\mathcal{O}(|G|^k)$ for storing three mTrees and their failure probability.

5.5.4. RegressionAdjustment *complexity*

Algorithm RegressionAdjustment has the same asymptotic complexity as MultinomialAlphaAdjustment. However, a significant reduction in computation time compared can be achieved when combining the two, instead of using MultinomialAlphaAdjustment only (see Fig. 4, green vs. orange line).

Algorithm 3: Algorithm RegressionAdjustment estimates the corrected significance level α_{corr} such that the mTree $m(\alpha_{corr}, k, p_G)$ has the probability of rejecting a fair ranking α

input : `kStart` – depth of the mTree to start with; k – the length of the ranking; p_G – the desired proportions of the protected groups; α – the desired significance level; `maxPreAdjustK` the maximum depth of the mTrees that are used as training data; `num_iterations` – the number of steps between `kStart` and `maxPreAdjustK`

output : α_{corr} – the adjusted significance

$R \leftarrow \{\}$; $\alpha_{new} \leftarrow \alpha$

// divide the interval [kStart,maxPreAdjustK] into num_iterations parts

`stepsize` $\leftarrow \max(\frac{\texttt{maxPreAdjustK}}{\texttt{num\_iterations}}, 1)$

for $i \leftarrow 0$ *to* `num_iterations` **do**

  // adjust α_{new} for the current kStart

  $\alpha_{\text{new}} \leftarrow$ MultinomialBinarySearchAdjustment(α_{new}, `kStart`, p_G)

  // add the pair (kStart, α_{new}) to the training data R

  R.*put*(`kStart`, α_{new})

  if `kStart` + `stepsize` $\leq$ `maxPreAdjustK` **then**

    `kStart` $\leftarrow$ `kStart` + `stepsize`

  end

  else

    break

  end

end

`coeffs` $\leftarrow$ R.*train*() // returns the vector of predicted coefficients for the curve over R

$\alpha_{c_r} \leftarrow$ `coeffs[0]` + `coeffs[1]` $* k +$ `coeffs[2]` $* k^2$

$\alpha_{corr} \leftarrow$ MultinomialBinarySearchAdjustment(α_{c_r}, k, p_G)

return α_{corr}

Table 3
Time complexity for all algorithms without pre-computed results.

Algorithm	Time complexity	Space complexity
InverseMultinomialCDF	$\mathcal{O}(\|G\|) \cdot \mathcal{O}(\text{MCDF}(k, p, \alpha))$	$\mathcal{O}(\|G\|^2)$
ComputeMTree	$\mathcal{O}(\|G\|^k) \cdot \mathcal{O}(\text{InverseMultinomialCDF})$	$\mathcal{O}(\|G\|^k)$
MultinomialAlphaAdjustment	$\mathcal{O}(\log \frac{\alpha}{\epsilon}) \cdot (\mathcal{O}(k^2) + \mathcal{O}(\text{ComputeMTree}))$	$\mathcal{O}(\|G\|^k)$
RegressionAdjustment	$\mathcal{O}(\log \frac{\alpha}{\epsilon}) \cdot (\mathcal{O}(k^2) + \mathcal{O}(\text{ComputeMTree}))$	$\mathcal{O}(\|G\|^k)$
FA*IR	$\mathcal{O}(n \log n) + \mathcal{O}(\text{MultinomialAlphaAdjustment}) + \mathcal{O}(k)$	$\mathcal{O}(\|G\|^k + n + k)$

5.6. Exact solution for the binomial case

In this section we described how to tackle the challenges of the inverse multinomial cumulative distribution function in the form of mTrees. However, the special case of two protected groups (the binomial case), which is extensively described by Zehlike et al. (2017), opens up a variety of more efficient and exact solutions. We developed an algorithm to compute the exact failure probability of a binomial mTree (mTable) and utilize it in a more efficient binary search for the correct significance level (Zehlike, Sühr & Castillo, 2020b). The implementation can be found in our GitHub repository.[3]

6. The multinomial FA*IR algorithm

In this section we present the multinomial FA*IR algorithm (Section 6.1) and determine its complexity (Section 6.2).

Algorithm 4: Algorithm FA*IR finds a ranking that maximizes utility subject to in-group monotonicity and ranked group fairness constraints. Checks for special cases (e.g., insufficient candidates of a class) are not included for clarity.

input : $k \in [n]$, the size of the list to return; $\forall\, c \in [n]$: q_c, the qualifications for candidate c, and g_c an indicator that is > 0 iff candidate c is protected; p_G with $\forall p_g \in (0, 1)$, the vector of minimum proportions for each group of protected elements; $\alpha_{corr} \in (0, 1)$, the adjusted significance for each fair representation test.
output : fair ranking τ^* satisfying the ranked group fairness condition (Def. ??).

$P_0, P_1, \dots P_{|G|} \leftarrow$ empty priority queues with bounded capacity k
for $c \leftarrow 1$ **to** n **do**
  insert c with value q_c in priority queue P_{g_c}
end
mtree$(i) \leftarrow$ ComputeMTree(k, p_G, α_{corr})
$(t_0, t_1, \dots, t_{|G|}) \leftarrow (0, \dots, 0)$
$i \leftarrow 0$
while $i < k$ **do**
  noCandidateAdded = True
  // get next node in tree path
  $m_{G,i} = [m_1(i), \dots, m_{|G|}(i)] \leftarrow$ findNextNode(mtree, i)
  // find which group needs a new candidate
  for *g = 1; g ≤ |G|; g++* **do**
    if $t_g < m_g(i)$ **then**
      // add a protected candidate
      $t_g \leftarrow t_g + 1$
      $\tau^*[i] \leftarrow \text{pop}(P_g)$
      noCandidateAdded = False
    end
  end
  if *noCandidateAdded* **then**
    // no protected candidate needed: add the best available
    $P_g \leftarrow$ findBestCandidateQueue()
    $\tau^*[i] \leftarrow \text{pop}(P_g)$
    $t_g \leftarrow t_g + 1$
  end
end
return τ^*

6.1. Algorithm description

Algorithm 4 presents Multinomial FA*IR, our solution to the Fair Top-k Ranking problem for multiple protected groups. The input of multinomial FA*IR is the target size k of the ranking to be produced, the scores or qualifications q_c, indicator variables g_c marking protected candidates, the minimum target probabilities p_G, and the test significance after adjustment α_{corr}.

[3] https://github.com/MilkaLichtblau/Multinomial_FA-IR.

The algorithm works as follows. In the initialization, qualifications q_c are used to create one ranked list of up to k candidates for each of the protected groups (P_g) and for the non-protected group (P_0). At Line 5, the algorithm creates a ranked group fairness tree (mTree) similar to the one shown on Fig. 3, indicating the minimum number of protected candidates needed at every position. Then, the algorithm greedily merges the per-group rankings, making sure to satisfy the minimum number of protected elements required at each position. If we require a protected candidate from group g, the algorithm picks the highest ranked element in P_g (Lines 12–16); otherwise, it picks the best candidate from $P_0 \cup P_1 \cup \cdots \cup P_{|G|}$ (Lines 18–22).

We observe that the choice of the group from which the element at position i should be drawn, i.e., the node selected at level i, depends on the path chosen along the `mtree`. In case there are multiple possibilities that satisfy ranked group fairness (as we saw in the example of Fig. 3 at level $k = 6$), the algorithm selects the branch that has the higher probability (larger F), breaking ties at random. Thus, the algorithm returns the most-likely child $m_{G,i}$ for a given parent (Line 10).

6.2. Algorithm complexity

Assuming a computational cost of $\mathcal{O}(n \log n)$ for creating the sorted lists of candidates, FA*IR ranks exactly k items using an adjusted mTree of height k. We need to run MultinomialAlphaAdjustment once and then follow one path through the mTree up to level k, leading to $\mathcal{O}(n \log n) + \mathcal{O}(\text{MultinomialAlphaAdjustment}) + \mathcal{O}(k)$. Note however, that if we used parameters (k, p_G, α) at any point in the past, we can obtain a previously calculated tree from disc. In this case FA*IR has a complexity of $\mathcal{O}(n \log n) + \mathcal{O}(k)$. The space complexity is $\mathcal{O}(n)$ for the candidates we want to rank, plus $\mathcal{O}(k)$ for the ranking itself, in summary $\mathcal{O}(n + k)$ (plus that of MultinomialAlphaAdjustment, if we have to calculate the mTree first, then leading to $\mathcal{O}(|G|^k + n + k)$). The complexity is summarized in Table 3.

6.3. Scalability and mTree reuse

In this section, we describe how FA*IR can be used in practice despite the high computational costs of the mTree calculation.

6.3.1. Storing and loading mTrees

Once computed, a mTree can be used either to test or to create fair rankings according to Definition 3.3. The mTree itself is independent of the data tested or ranked and thus is reusable for all rankings of length $m \leq k$ with required proportions p_G and significance α. Since testing and ranking with mTrees is efficient, but mTree creation is not, it is important to store already computed mTrees and never compute a mTree twice. Our implementation[4] provides the infrastructure for loading and storing mTrees for all steps of our algorithm. We also provide the pre-computed set of mTrees from our experiments.

6.3.2. Reusing mtrees

As mentioned above, a mTree of depth k can be used to create and test rankings of every length up to and including k. Thus, for given α, p_G only the largest mTree has to be stored. Furthermore, all mTrees calculated during the binary search in Algorithm 3 MultinomialAlphaAdjustment can be reused too, if only the parameter α changes. More specifically, MultinomialAlphaAdjustment computes many mTrees which are persisted. If we want to later compute a mTree for the same p_G but with a different α, many of the binary search steps (computed mTrees) will be repeated and the respective trees can be loaded from disc. In summary, the first calculation of a large mTree for many protected groups is expensive, but has to be done only once. After that, adjustments of k and α are fast through recycling of already computed mTrees and MCDFs (see also Section 5.4). The only case in which mTrees cannot be reused easily is if p_G changes. However, in many cases p_G is determined by law, or alternatively by policies or guidelines that change in a time frame of years. For instance, the International Labour Organization's research shows that, among the countries that seek to increase the participation of people with disabilities in employment, most set a fixed quota between 1% and 10%; some countries also set an intersectional quota for people with disabilities (International Labor Organization, 2019). The European Union has goals of 40% of participation for women in several areas, including company boards, but also slates of candidates for elected positions.[5] Under those circumstances, p_G do not change as often as other elements, such as the value of k used to display information to users.

6.4. Partial ordering of solutions

When there is a single protected group, top-k rankings that have a given utility form a total ordering with respect to group fairness, and top-k rankings that have a given ranked group fairness form a total ordering with respect to utility. This allowed us to prove that FA*IR for a single protected group finds the optimal solution w.r.t. selection and ordering utility (Zehlike et al., 2017). The proof comes from the observation that decreases in utility that maintain ranked group fairness can only occur when a high-scoring non-protected candidate is ranked below a low-scoring protected one. When multiple groups are present there is no total ordering as some rankings cannot be compared. It may happen that a high-scoring candidate from a *better-performing protected* group is ranked below a low-scoring candidate from a *low-performing protected* group. This means that ordering inversions can no longer happen only w.r.t. the non-protected group, but also among the protected groups. As the ranked group fairness criterion for

[4] https://github.com/MilkaLichtblau/Multinomial_FA-IR.

[5] https://www.theguardian.com/world/2020/mar/05/eu-revives-plans-for-mandatory-quotas-of-women-on-company-boards.

Table 4
Datasets and experimental settings. The ad-hoc score for COMPAS was calculated by a weighted summation of recidivism risk, number of prior arrests and violent recidivism risk.

	Dataset	n	k	Qualification criterion	Protected groups	Protected %
D1	COMPAS Angwin et al. (2016)	6173	1500	ad-hoc score	People of color (PoC)	65.9%
D2	COMPAS Angwin et al. (2016)	6173	500	ad-hoc score	25 yr. < x < 45 yr. <25 yr.	57.2% 21.8%
D3	COMPAS Angwin et al. (2016)	6173	300	ad-hoc score	PoC female, <25 yr. White female, < 25 yr. PoC male, <25 yr.	2.8% 1.2% 13.4%
D4	German credit Lichman (2013)	1000	50	Credit rating	Female, non-prot. age Male, oldest 10% Male, youngest 10% Female, oldest 10% Female, youngest 10%	21.7% 6.3% 4.4% 3.2% 6.1%
D5	LSAT Wightman and Ramsey (1998)	21 K	300	LSAT score	White, female PoC, female PoC, male	35.3% 8.4% 7.6%

multiple groups can be expressed as a tree-structure and *any* path satisfies the constraint, we have multiple choices of which path to choose to satisfy the same ranked group fairness. Hence, multiple solutions (rankings) are equivalent.

The way in which we decided to choose among these equivalent rankings is in line with the probabilistic nature of the design of FA*IR. We decided to pick the path that is "most likely", *i.e.*, that has the highest MCDF value. An alternative implementation could select the highest utility path, picking the candidate with the highest qualifications when multiple choices exist. This would avoid having to rank a protected candidate with lower utility above a protected candidate with a higher utility from another protected group. However, that would require to assume that measures of qualification are comparable across groups of candidates. This would be a strong assumption due to different manifestations of bias in score distributions seen in practice. Scores have been even proven to be biased against protected groups, as is the case with the COMPAS scores (Angwin et al., 2016) that we use in the experiments of Section 7, but this bias manifests differently across groups (as an example see Fig. 5(c)). Let us consider an example to clarify this point. Suppose that we had two protected groups, namely white female and black female candidates, and all other candidates were non-protected. Suppose that the degree of bias in the scores for black females is substantially higher than for white females, i.e., black females obtain lower scores for the same performance as white females. This is a plausible scenario due to the intersectional discrimination which black females face. In this case, choosing among equivalent solutions, the mTree path that maximizes utility would result in unfairly preferring white females over black females. Instead, choosing the most likely path follows a probabilistic design, and only requires that scores are comparable within groups, but not across.

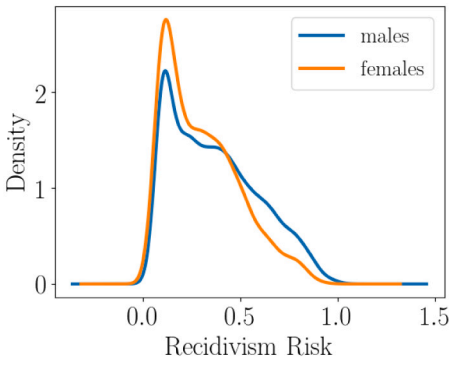
7. Experiments

In this section, we evaluate the multinomial FA*IR algorithm. We present the datasets we used in Section 7.1, metrics and comparison with baselines in Section 7.2, and results in Section 7.3.

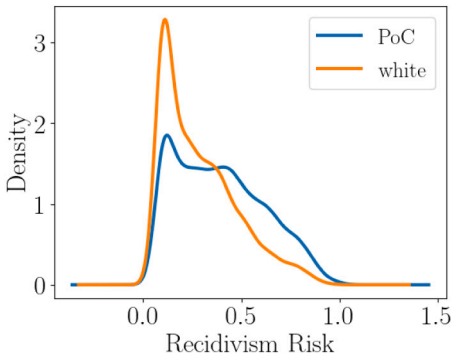
7.1. Datasets

We used multiple datasets in our experiments, each dataset represents a set of people with certain demographic attributes, and includes a "score" attribute. Table 4 presents some general characteristics of each dataset. Dataset D1 corresponds to the same experimental setup as in our previous work (Zehlike et al., 2017), but instead of using the original algorithm FA*IR, we run the binomial experiment with the multinomial extension proposed in Section 3. We perform tests with various values of $k < n$ in each dataset. Additionally, we tested different definitions of what would constitute a protected group within each dataset, for experimental purposes. However, we must note that in a real application there is usually a protected group that is clearly defined by voluntary agreements or by law, and that group historically and currently experiences a disadvantage. Similarly, the minimum proportion can always be provided by regulation or voluntary commitments; in the absence of this minimum proportion one can use the fraction of people who are members of that group within the entire population. An experiment consists of generating a ranking from the set of candidates using multinomial FA*IR and then comparing it with baseline rankings according to the metrics introduced in the next section. We use the three publicly-available datasets: COMPAS (Angwin et al., 2016), German Credit (Lichman, 2013) and LSAC (Wightman & Ramsey, 1998).

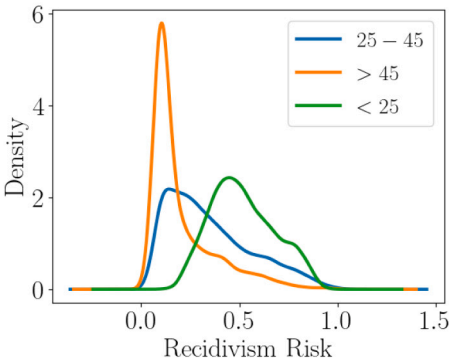
COMPAS is a criminal recidivism risk assessment instrument that predicts the probability of a convict to recidivate within the two years after the sentence. It is based on a set of over a hundred items or questions, and currently it is used by several jurisdictions in the US. COMPAS has been accused of racial discrimination by having a larger false positive rate for African Americans (Angwin et al., 2016). In our experiment, we test a scenario in which we want to create a fair ranking of the top-k people who are least likely to recidivate, who could be, for instance, considered for an alternative program to prison. We calculated a candidate's overall score



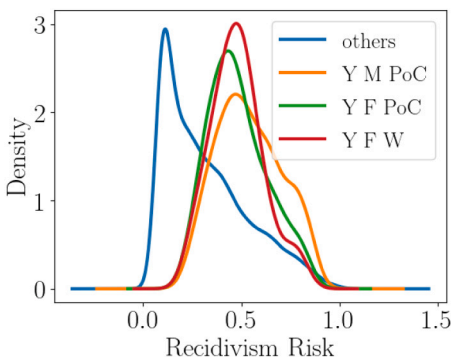
(a) Score distribution of male and female candidates in the COMPAS dataset.



(b) Score distribution of white candidates and people of color (PoC) in the COMPAS dataset. Experiment D1.



(c) Score distribution by age in the COMPAS dataset. We see that age is a strong predictor of recidivism risk which suggests that the COMPAS questionnaire injects a bias against younger people into the data. Experiment D2.



(d) Score distribution protected and non-protected candidates in the COMPAS dataset. Protected groups are young male PoC, young female PoC and young white females, which are the three groups with lowest average exposure in the colorblind ranking. Experiment D3.

Fig. 5. Distribution of COMPAS scores in different experimental settings. The first three figures show distributions separately by each of the three protected categories. The fourth figure shows the distribution for dataset D3.

as a weighted summation of the columns "recidivism, violent recidivism" and "prior arrests" from the original dataset. To break ties, we added normally distributed random noise with $\mu = 0, \sigma = 0.00001$ to the score of each candidate. Our protected groups are formed by different combination of the attributes "race, age" and "sex", where race is either "white" or "person of color" (PoC), age is either "younger than 25", "between 25 and 45" or "older than 45", and sex is either "male" or "female". We observe in Fig. 5(d) that PoC 5(b), as well as males 5 are given a larger recidivism score than other groups. However we see in Fig. 5(c) that the protected attribute "age" has the strongest impact on recidivism risk. Apparently COMPAS tends to give larger risk scores to younger people. We therefore consider white, female and older than 45 as the *non-protected* categories for our experiments.

In dataset D1 we consider people of color as the protected group and conduct experiments with two different minimum proportion vectors. The first one sets the p-values to match the protected group proportion in the dataset $p_{\text{stat}} = [0.66]$ which relates to group fairness as statistical parity; the second one sets all minimum proportions to the same value $p_{\text{eq}} = [0.5]$.

In dataset D2 we consider people aged 25–45 and younger than 25 as protected and the vectors p_G are $p_{\text{stat}} = [0.573, 0.218]$ and $p_{\text{eq}} = [0.333, 0.333]$.

In dataset D3 we divided the candidates into 12 different groups that we constructed from the Cartesian product of all three protected categories "age", "race", and "sex". Then we calculated the average group exposure for each group and considered as protected the three groups that were placed lowest in the colorblind ranking. These are young female persons of color, young white females and young male PoC. Interestingly, despite the fact that females have higher scores on average, two of the three groups with lowest exposure values are female (Fig. 5(d)). We set the vectors p_G to $p_{\text{stat}} = [0.028, 0.012, 0.134]$ and $p_{\text{eq}} = [0.25, 0.25, 0.25]$.

German Credit is a dataset based on ratings generated by the German agency *Schufa,* and is also known as the Statlog German Credit Dataset. The score given by *Schufa* is based on age, gender, marital status, and other features of each applicant, and is widely

used in Germany as a credit rating score for operations such as renting an apartment or applying for a loan. The dataset provides a credit-worthiness score that we use as "qualification", and corresponds to a weighted sum of credit duration, credit amount, and employment length. As protected attributes we use the sex and age of a candidate: females and whether or not they belong to the group of the 100 youngest or oldest persons respectively form the protected groups, because these tend to be given lower scores. We set the vectors p_G to $p_{\text{stat}} = [0.217, 0.063, 0.044, 0.032, 0.061]$ and $p_{\text{eq}} = [0.166, 0.166, 0.166, 0.166, 0.166]$.

LSAT is a dataset collected by Wightman and Ramsey (1998) to study whether the admission metrics to law schools in the US have a disparate impact on students of color. The qualification attribute consists of scores in the US Law School Admission Test (LSAT). The protected features are a person's sex and whether or not they belong to the group of people of color (PoC). Women and PoC score on average lower than men and Whites in this test, which is why we regard white men as non-protected and the other groups as protected. We set the vectors p_G to $p_{\text{stat}} = [0.353, 0.084, 0.076]$ and $p_{\text{eq}} = [0.25, 0.25, 0.25]$.

7.2. Baseline methods and metrics

The baseline methods for comparison that we use are the following:

Baseline 1: Group-unaware or "color-blind" ranking. This method merely order the candidates by decreasing scores/qualifications (see Section 3.1) and therefore maximizes selection and ordering utility. We evaluate all methods relative to the metrics achieved by the colorblind ranking, i.e., we report utility loss and fairness gain of the methods w.r.t. the colorblind ranking.

Baseline 2: Continuous Fairness Algorithm (CFAθ). This is a post-processing method introduced by Zehlike, Hacker, et al. (2020), and as explained in Section 2.2, to the best of our knowledge this is the only utility-based method in the literature that does not assume that utilities can be compared across groups. It aligns the score distributions of the protected candidates with the Wasserstein-barycenter of all group distributions. To achieve this the algorithm finds a new distribution of scores for each group, the "fair representation", by interpolating between the barycenter and the group distribution, subject to a given fairness parameter θ. This fair representation corresponds to the idea to rank groups separately and then subsequently pick the best candidates from each group to create the result ranking. Note that setting $\theta = 1$ to its maximum corresponds to setting the minimum proportions in p_G to the dataset proportions of each group. This means that with CFAθ we cannot achieve an exposure gain that would require minimum proportions beyond statistical parity. Hence this method is comparable to our method only if all values in p_G are less or equal to statistical parity. In our experiments we therefore only compare FA*IR with p_{stat} to CFAθ with $\theta = 1$, as higher p-values are not comparable.

Baseline 3: Categorical sampling or "dice roll". We adapt the Bernoulli process of Yang and Stoyanovich (2016) to a multinomial setting, in which more than one protected group is present. This corresponds to sequentially rolling a $|G|$-sided dice for each of the ranking positions $i = 1, 2, \ldots, k$, placing at ranking position i the best available candidate from the group whose side is showing (with each candidate appearing at most once in the ranking). Each side of the dice represents one group $g \in G$ and shows with probability p_g. We note that this procedure, if repeated many times, will approximate the results of FA*IR, and we report mean and standard deviation of the metrics for 10,000 rankings created through the dice roll procedure. The difference is that our method is deterministic and therefore guarantees a certain share of visibility for each group for each created ranking.

Baseline 4: Binomial FA*IR. We additionally report experimental results using the binomial version of FA*IR (Zehlike et al., 2017; Zehlike, Sühr, Castillo, & Kitanovski, 2020). Because this version of FA*IR accepts only one value $p \in \mathbb{R}$, we aggregated all protected groups into one. The evaluation however, is shown for each group separately to depict the differences between the binomial and the multinomial version of FA*IR. The resulting values for p are $p = \sum p_{\text{eq}}$ which is the sum of the entries of p_G, where all groups have the same minimum proportion. In the second setting, p is calculated as $p = \sum p_{\text{stat}}$ which is the sum of the entries of p_G, where all groups have minimum proportions matching their respective dataset shares.

Utility (Performance Measure). We report the loss in ranked utility after score normalization, in which all q_i are normalized to be within $[0, 1]$. We also report the maximum rank drop, *i.e.*, the number of positions lost by the candidate that realizes the maximum ordering utility loss.

NDCG (Performance Measure). We report a normalized weighted summation (Normalized Discounted Cumulative Gain — NDCG) of the scores of the elements in the ranking, $\sum_{i=1}^{k} w_i q_{(\tau_i)}$, in which the weights are chosen to have a logarithmic discount in the position: $w_i = \frac{1}{\log_2(i+1)}$. This is a standard measure to evaluate search rankings (Järvelin & Kekäläinen, 2002). This is normalized so that the maximum value is 1.0.

Kendall's Tau (Performance Measure). We report the similarity of the fair candidate orderings w.r.t. the colorblind ranking in terms of Kendall's τ correlation coefficient. A value close to 1 indicates strong similarity between the two rankings (with 1 meaning perfect agreement), a value close to -1 indicates strong dissimilarity (with -1 meaning perfectly reversed ordering), and a value of 0 indicates there is no correlation between the two rankings.

Exposure (Fairness Measure). We use a measure of average group exposure, which we define as the average position bias that a group is exposed to. In rankings, their exposure to the user is critical for ranked candidates to benefit from the system and if a group of candidates is systematically ranked low, it can be considered as biased (Friedman & Nissenbaum, 1996). We model position bias v by means of a logarithmic progression of the form $v(k) = \frac{1}{\log_2(i+1)}$. Thus the first position has a bias $v(1) = 1$, which then decreases logarithmically. Higher position bias translates into more exposure and hence more visibility. We show that large increases in average group exposure can be achieved with relatively small losses in ordering utility and NDCG.

Table 5
Experimental results, showing changes in the average group exposure and ranking utility in terms of NDCG. Exposure is presented per group while the loss of NDCG is calculated for the entire ranking. The protected groups are in the same order as in Table 4. For exposure gain we report numbers with an asterisk, if a group was not among the top-k in the colorblind ranking, but is now. These numbers constitute the *absolute* exposure value a group receives after re-ranking. We report mean and standard deviation of all metrics for the dice roll baseline. Note however, that these are just result statistics which do not relate to a specific ranking. Comparable results are grouped together.

Experiment	Method	Exposure gain per prot. group	Exposure gain non-prot. group	Total NDCG loss	Kendall's Tau
D1 — COMPAS, k = 1500, race (1 prot.)	mult. FA*IR p_{eq}	0.25	−0.25	0.0000	0.99
	dice roll p_{eq}	−0.45	0.45	0.0055	0.98
	dice roll p_{eq} mean/std	−1.74/2.32	1.74/2.32	0.0002/0.0002	0.97/0.01
	binom. FA*IR $p = \sum p_{\text{eq}}$	0.00	0.00	0.0000	1.00
	mult. FA*IR p_{stat}	19.77	−19.77	0.0040	0.82
	dice roll p_{stat}	28.47	−28.47	0.0070	0.77
	dice roll p_{stat} mean/std	25.73/2.22	−25.73/2.22	0.0057/0.0009	0.78/0.02
	CFAθ	28.16	−28.16	0.0050	0.66
	binom. FA*IR $p = \sum p_{\text{stat}}$	18.75	−18.75	0.0037	0.83
D2 — COMPAS, k=500, age (2 prot.)	mult. FA*IR p_{eq}	[−7.34, 20.87*]	−13.53	0.0580	0.55
	dice roll p_{eq}	[−5.07, 21.72*]	−16.64	0.0585	0.51
	dice roll p_{eq} mean /std	[−6.67/1.62, 23.5*/1.63]	−16.83/1.62	0.0655/0.0054	0.49/0.03
	binom. FA*IR $p = \sum p_{\text{eq}}$	[12.55, 0.00*]	−12.55	0.0004	0.76
	mult. FA*IR p_{stat}	[7.95, 14.11*]	−22.06	0.0354	0.51
	CFAθ	[24.33, 15.91*]	−38.48	0.0380	0.37
	dice roll p_{stat}	[8.83, 15.68*]	−24.52	0.0396	0.47
	dice roll p_{stat} mean/std	[10.23/1.71, 15.38*/1.42]	−25.61/1.39	0.0387/0.0042	0.48/0.03
	binom. FA*IR $p = \sum p_{\text{stat}}$	[21.62, 0.00*]	−21.62	0.0011	0.74
D3 — COMPAS, k=300, 3 worst off (3 prot.)	mult. FA*IR p_{eq}	[10.30*, 10.18*, 10.25*]	−30.72	0.1808	0.38
	dice roll p_{eq}	[11.60*, 11.81*, 11.81*]	−35.21	0.2083	0.29
	dice roll p_{eq} mean	[11.85*, 11.34*, 11.84*]	−35.03	0.2088	0.33
	dice roll p_{eq} std	[1.31, 0.86, 1.30]	1.29	0.0094	0.05
	binom. FA*IR $p = \sum p_{\text{eq}}$	[6.42*, 2.22*, 22.77*]	−31.40	0.1535	0.61
	mult. FA*IR p_{stat}	[0.84*, 0.41*, 4.08*]	−5.33	0.0180	0.81
	CFAθ	[0.93*, 0.24*, 6.74*]	−7.92	0.0236	−0.12
	dice roll p_{stat}	[2.03*, 0.59*, 7.01*]	−9.73	0.0353	0.69
	dice roll p_{stat} mean	[1.33*, 0.57*, 6.34*]	−8.24	0.0292	0.71
	dice roll p_{stat} std	[0.49, 0.33, 1.02]	1.15	0.0046	0.04
	binom. FA*IR $p = \sum p_{\text{stat}}$	[1.11*, 0.63*, 3.82*]	−5.56	0.0187	0.80
D4 — German cred., k=50, sex & age (5 prot.)	mult. FA*IR p_{eq}	[0.08, 0.50, 1.49, 1.26, 1.12]	−4.46	0.1685	0.51
	dice roll p_{eq}	[−1.09, −0.13, 2.76, 1.28, 1.75]	−4.58	0.2315	0.36
	dice roll p_{eq} mean	[−0.31, 0.45, 1.00, 1.82, 1.74]	−4.70	0.2265	0.47
	dice roll p_{eq} std	[0.77, 0.78, 0.77, 0.77, 0.77]	0.76	0.0449	0.08
	binom. FA*IR $p = \sum p_{\text{eq}}$	[2.41, 0.40, 0.00, 0.36, 0.25]	−3.42	0.0619	0.83
	mult. FA*IR p_{stat}	[0.01, 0.23, 0.38, 0.20, 0.01]	−0.83	0.0185	0.84
	CFAθ	[0.34, −0.93, −0.66, −0.03, 0.28]	1.00	0.0426	0.74
	dice roll p_{stat}	[0.89, −1.24, −0.38, −0.33, 0.26]	0.80	0.0560	0.66
	dice roll p_{stat} mean	[0.37, −0.89, −0.55, 0.05, 0.38]	0.64	0.0675	0.65
	dice roll p_{stat} std	[0.86, 0.51, 0.43, 0.35, 0.50]	1.02	0.0209	0.65
	binom. FA*IR $p = \sum p_{\text{stat}}$	[0.00, 0.00, 0.00, 0.00, 0.00]	0.00	0.0000	1.00
D5 — LSAT, k=300, sex & race (3 prot.)	mult. FA*IR p_{eq}	[−0.32, 10.71, 10.96]	−21.35	0.0207	0.08
	dice roll p_{eq}	[−6.50, 9.19, 14.69]	−17.47	0.0263	0.07
	dice roll p_{eq} mean	[−5.03, 10.61, 10.69]	−16.27	0.0211	0.06
	dice roll p_{eq} std	[1.31, 1.30, 1.30]	1.31	0.0021	0.07
	binom. FA*IR $p = \sum p_{\text{eq}}$	[12.94, 1.56, 1.20]	−15.71	0.0048	0.14
	mult. FA*IR p_{stat}	[4.00, 4.70, 4.89]	−13.58	0.0053	0.09
	CFAθ	[−0.74, 1.27, 4.64]	−5.18	0.0009	−0.03
	dice roll p_{stat}	[−0.16, 3.36, 2.24]	−5.44	0.0028	0.23
	dice roll p_{stat} mean	[−0.25, 2.71, 2.43]	−4.89	0.0023	0.14
	dice roll p_{stat} std	[1.44, 0.83, 0.80]	1.51	0.0007	0.10
	binom. FA*IR $p = \sum p_{\text{stat}}$	[5.35, 0.54, 0.44]	−6.33	0.0007	0.31

7.3. Results

Tables 5 and 6 summarize the results. We report on the result for FA*IR using p_G in two different settings, namely as a statistical parity vector i.e. the p-values for each group correspond to their respective proportions in the dataset (p_{stat}), and as a vector with all p-values being equal (p_{eq}). We report experimental results for the dice rolling baseline using same probability vectors p_G as for FA*IR. The actual values for p_G are given in the dataset descriptions in Section 7.1. Also remember that results of CFAθ are only

Table 6
Experimental results, changes of individual utility loss respect to colorblind results. All measures are presented per group. Groups are in the same order as in Table 4, and the first value is always for the non-protected group. We report mean and standard deviation of all metrics for the dice roll baseline. Note however, that these are just result statistics which do not relate to a specific ranking. Comparable results are grouped together.

Experiment	Method	Ordering utility loss	Rank drop	Selection utility loss
D1 — COMPAS, k = 1500, race (1 prot.)	mult. FA*IR p_{eq}	[0.0066, 0]	[17, 0]	[0, 0]
	dice roll p_{eq}	[0.0066, 0.0066]	[60, 30]	[0, 0.0066]
	dice roll p_{eq} mean/std	[0.0061/0.0061, 0.0135/0.0089]	[32.19/19.38, 42.15/26.21]	[0.0002/0.0014, 0.0118/0.0095]
	binom. FA*IR $p = \sum p_{eq}$	[0, 0]	[0, 0]	[0, 0]
	mult. FA*IR p_{stat}	[0.0702, 0]	[387, 0]	[0.0621, 0]
	CFAθ	[0.0687, 0]	[579, 297]	[0.0636, 0]
	dice roll p_{stat}	[0.0833, 0]	[524, 0]	[0.0833, 0]
	dice roll p_{stat} mean/std	[0.0772/0.0056, 0/0]	[477.35/36.78, 0.45/1.10]	[0.0736/0.0078, 0/0]
	binom. FA*IR $p = \sum p_{stat}$	[0.0636, 0]	[373, 0]	[0.0621, 0]
D2 — COMPAS, k=500, age (2 prot.)	mult. FA*IR p_{eq}	[0.23, 0, 23, 0.00]	[182, 116, 0]	[0.23, 0.23, 0.00]
	dice roll p_{eq}	[0.23, 0.23, 0.00]	[214, 77, 0]	[0.23, 0.23, 0.00]
	dice roll p_{eq} mean/std	[0.24/0.01, 0.24/0.01, 0/0]	[214.5/17.0, 103.4/20.9, 0/0]	[0.24/0.01, 0.24/0.01, 0/0]
	binom. FA*IR $p = \sum p_{eq}$	[0.01, 0, 0]	[180, 0, 0]	[0.01, 0, 0]
	mult. FA*IR p_{stat}	[0.21, 0.21, 0.00]	[287, 0, 0]	[0.21, 0.21, 0.00]
	CFAθ	[0.19, 0.18, 0.00]	[159, 282, 0]	[0.19, 0.17, 0.00]
	dice roll p_{stat}	[0.22, 0.22, 0.00]	[310, 0, 0]	[0.22, 0.22, 0.00]
	dice roll p_{stat} mean/std	[0.21/0.01, 0.21/0.01, 0/0]	[323.4/16.2, 1.4/1.9, 0/0]	[0.22/0.01, 0.21/0.01, 0/0]
	binom. FA*IR $p = \sum p_{stat}$	[0.01, 0, 0]	[283, 0, 0]	[0.01, 0, 0]
D3 — COMPAS, k=300, 3 worst off (3 prot.)	mult. FA*IR p_{eq}	[0.60, 0.26, 0.00, 0.37]	[206, 0, 0, 0]	[0.60, 0.26, 0.00, 0.37]
	dice roll p_{eq}	[0.63, 0.37, 0.00, 0.49]	[209, 0, 0, 0]	[0.71, 0.37, 0.00, 0.49]
	dice roll p_{eq} mean	[0.67, 0.33, 0.00, 0.44]	[218.5, 0, 0, 0]	[0.67, 0.32, 0.00, 0.45]
	dice roll p_{eq} std	[0.08, 0.08, 0.01, 0.08]	[7.9, 0, 0, 0]	[0.07, 0.08, 0.00, 0.07]
	binom. FA*IR $p = \sum p_{eq}$	[0.28, 0, 0, 0]	[208, 0, 0, 0]	[0.28, 0, 0, 0]
	mult. FA*IR p_{stat}	[0.17, 0.03, 0.00, 0.02]	[38, 0, 0, 0]	[0.17, 0.01, 0.01, 0.00]
	CFAθ	[0.15, 0.01, 0.00, 0.00]	[268, 0, 0, 0]	[0.15, 0.00, 0.00, 0.00]
	dice roll p_{stat}	[0.20, 0.001, 0.006, 0.02]	[62, 0, 0, 0]	[0.20, 0.01, 0.03, 0.00]
	dice roll p_{stat} mean	[0.20, 0.03, 0.02, 0.04]	[52.0, 0, 0, 0]	[0.20, 0.02, 0.02, 0.00]
	dice roll p_{stat} std	[0.01, 0.02, 0.01, 0.03]	[6.5, 0, 0, 0]	[0.01, 0.02, 0.01, 0.01]
	binom. FA*IR $p = \sum p_{stat}$	[0.17, 0, 0, 0]	[39, 0, 0, 0]	[0.17, 0, 0, 0]
D4 — German cred., k=50, sex & age (5 prot.)	mult. FA*IR p_{eq}	[3.57, 2.70, 3.42, 0.55, 1.23, 1.76]	[28, 4, 6, 1, 0, 0]	[3.05, 1.82, 1.34, 0.00, 0.18, 0.33]
	dice roll p_{eq}	[3.65, 5.07, 4.19, 1.05, 1.23, 2.14]	[22, 17, 9, 0, 0, 0]	[3.13, 2.27, 1.46, 0.00, 0.25, 0.23]
	dice roll p_{eq} mean	[3.32, 3.00, 3.00, 2.65, 1.20, 1.14]	[26.2, 7.9, 11.8, 10.5, 1.7, 0.1]	[3.13, 1.99, 1.38, 0.12, 0.20, 0.31]
	dice roll p_{eq} std	[0.36, 0.85, 0.76, 0.95, 0.90, 0.68]	[5.3, 6.7, 9.3, 8.8, 3.9, 0.8]	[0.25, 0.43, 0.51, 0.38, 0.30, 0.38]
	binom. FA*IR $p = \sum p_{eq}$	[1.53, 0, 0, 0, 0, 0]	[25, 0, 0, 0, 0, 0]	[1.37, 0, 0, 0, 0, 0]
	mult. FA*IR p_{stat}	[1.84, 1.56, 0.45, 0.00, 0.33, 1.28]	[5, 3, 1, 0, 0, 5]	[1.89, 1.56, 1.23, 0.00, 1.18, 1.54]
	CFAθ	[0.50, 0.91, 2.07, 1.46, 0.72, 0.00]	[2, 1, 32, 14, 2, 0]	[0.34, 0.00, 1.93, 2.20, 0.00, 0.00]
	dice roll p_{stat}	[0.92, 0.54, 1.76, 2.55, 0.00, 0.29]	[1, 0, 15, 38, 0, 0]	[0.96, 0.56, 2.74, 0.00, 2.92, 0.20]
	dice roll p_{stat} mean	[1.23, 1.59, 2.15, 2.03, 1.11, 0.46]	[3.7, 4.6, 24.7, 23.1, 10.5, 1.8]	[0.85, 0.49, 1.77, 1.71, 0.75, 0.34]
	dice roll p_{stat} std	[0.53, 0.81, 0.81, 0.96, 0.95, 0.56]	[3.6, 4.8, 11.9, 14.0, 12.7, 4.3]	[0.47, 0.49, 1.15, 1.38, 1.11, 0.51]
	binom. FA*IR $p = \sum p_{stat}$	[0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 0, 0]
D5 — LSAT, k=300, sex & race (3 prot.)	mult. FA*IR p_{eq}	[4, 4, 1, 0]	[220, 10, 0, 0]	[4, 3, 0, 0]
	dice roll p_{eq}	[5, 5, 2, 0]	[194, 81, 0, 0]	[5, 5, 1, 0]
	dice roll p_{eq} mean	[4.2, 4.2, 1.4, 0.3]	[180.7, 69.8, 0, 0]	[4.2, 4.2, 0.3, 0.02]
	dice roll p_{eq} std	[0.4, 0.4, 0.5, 0.5]	[10.5, 16.7, 0, 0]	[0.4, 0.4, 0.5, 0.1]
	binom. FA*IR $p = \sum p_{eq}$	[1, 0, 0, 0]	[157, 0, 0, 0]	[1, 0, 0, 0]
	mult. FA*IR p_{stat}	[2, 2, 1, 0]	[136, 13, 0, 0]	[2, 1, 0, 0]
	CFAθ	[1, 1, 1, 0]	[122, 26, 0, 0]	[1, 0, 0, 0]
	dice roll p_{stat}	[2, 2, 1, 0]	[75, 12, 0, 0]	[2, 1, 0, 0]
	dice roll p_{stat} mean	[2.0, 1.9, 0.8, 0.4]	[64.2, 16.1, 0.3, 1.1]	[2.0, 1.2, 0.5, 0.04]
	dice roll p_{stat} std	[0.4, 0.5, 0.5, 0.5]	[15.3, 11.0, 2.5, 5.7]	[0.4, 0.6, 0.6, 0.2]
	binom. FA*IR $p = \sum p_{stat}$	[1, 0, 0, 0]	[113, 0, 0, 0]	[1, 0, 0, 0]

comparable to a FA*IR p_{stat} setting. We therefore group those results together that allow for a meaningful comparison. With the p_{eq} setting, we illustrate the flexibility of our approach compared to CFAθ, which can achieve statistical parity at maximum.

First, we observe that, for FA*IR p_{stat} and CFAθ, in general changes in NDCG with respect to the color-blind ranking are minor. This can be explained as the utility is dominated by the top positions, which usually do not change dramatically. However, we see that FA*IR outperforms CFAθ significantly in terms of Kendall's tau, which means that the rankings constructed by CFAθ have much more ordering inversions w.r.t. the color-blind ranking than those constructed by FA*IR.

Second we see that in experiments D1 and D5 FA*IR with p_{stat} and the baseline CFAθ seem to perform more or less equally well at first glance. While the former yields better numbers in terms of utility, with the latter the protected groups gain higher exposure.

Nonetheless, we found a very important drawback in the results of CFAθ in experiment D2: Zehlike, Hacker, et al. (2020) does not guarantee in-group monotonicity, which we observed in our experiments. Consider Table 6 where we observe a maximum rank drop for the non-protected group of 159 for CFAθ, which at first glance is less than FA*IR with p_{stat} (287). However, we found the following problems when closely investigating the result: the candidate that experienced the drop in 159 ranks was the *only non-protected* candidate left in the top-500 after applying CFAθ, and their original ranking position was 341 (now 500). This means that the originally best candidate from the non-protected group (rank 3 in $cb|_k$) was ranked out of the top-500 by CFAθ, hence the in-group ordering is not preserved. We suspect that this is because the method cannot handle ties well. If lots of candidates have very similar original scores, the fair representation scores may have large ties in which candidates are ordered randomly and not w.r.t. their original ordering. In such a setting, FA*IR has an important advantage: it guarantees in-group monotonicity, because candidates are separated into group rankings which are sorted by decreasing *original* qualifications. No matter which group has to be picked at the current position, the respective candidate will always be the best available from that group. This means that while very similar scores appear to be problematic for CFAθ, they are the best-scenario for FA*IR, because ranked-group fairness can be ensured without losing any relevance at all (NDCG stays unaffected). This problem also explains the large differences between the methods in terms of Kendall's Tau.

Third we observe a formerly known weakness of CFAθ manifesting in experiment D4: it cannot handle datasets with very small groups or groups that show low variance. It increases the exposure of the non-protected group while decreasing it for the oldest males, and youngest males and females. In contrast, FA*IR p_{stat} can handle very small groups, and gives them sufficient exposure that is in line with the minimum proportions p_G. For experiment D4 we observe an interesting advantage of FA*IR: the youngest and oldest males experience a high rank drop (31 and 22) with CFAθ compared to FA*IR p_{stat}, even though they belong to the protected groups. This is because they receive exposure values higher than average in the $cb|_k$, i.e. these two groups have the best scores. CFAθ moves *all* score distributions towards the barycenter and thus, removes the advantage that the protected groups have in this dataset. FA*IR, in contrast, ranks candidates based on their scores, as soon as the ranked group fairness conditions are met for each group. As such, a protected group can only lose exposure for another protected group, but never for the non-protected one.

When considering the categorical sampling or "dice rolling" baseline, we see our expectations of non-consistent behavior confirmed. This can be explained by the involved randomness of the dice rolling procedure. We observe that in experiments D2 and D3 FA*IR p_{eq} and *dice roll* p_{eq} yield very similar results. However in experiments D4 and D5 FA*IR p_{stat} and *dice roll* p_{stat} yield very different results, even though the underlying stochastic process of the two methods is the same. In contrast to a dice rolling process, FA*IR constructs fair rankings in a deterministic way and therefore always guarantees a minimum representation for each protected group.

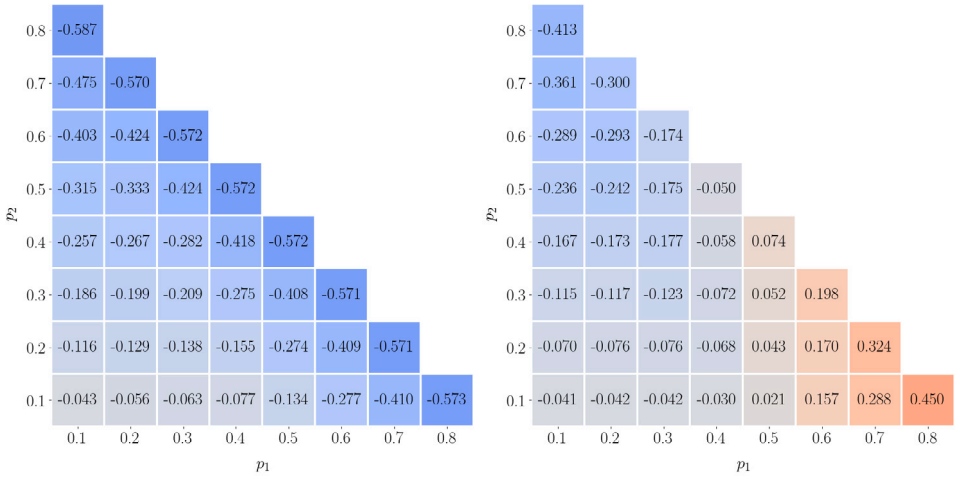
Comparison of Binomial FA*IR and Multinomial FA*IR. Tables 5 and 6 reveal the importance of considering multiple protected groups and thus using the multinomial version of FA*IR instead of the binomial version. Consider D3 — COMPAS, where binomial FA*IR achieves higher Kendall's Tau and less total NDCG loss. However, when we look at the exposure gain per group, we can see that this is due to less improvements of protected group 1 and 2 compared to the multinomial version of FA*IR. As we explained in the introductory example, binomial FA*IR will order all candidates from the artificially created protected group by decreasing qualifications and thus rank less candidates of a real protected group to higher positions, if they show low overall scores. This effect can lead to no improvement for certain protected groups at all, as in D2 — COMPAS for group 2, and D4 — German credit for group 2. D4 — German credit reveals another weakness of binomial FA*IR compared to multinomial FA*IR: For $p = \sum p_{\text{stat}}$ the colorblind ranking is already "binomial"-fair, while the multinomial version of FA*IR improves visibility for all protected groups according the respective minimum proportions.

Having said that, D1 — COMPAS reveals a strength of the binomial FA*IR implementation: though we would not expect to see any difference between multinomial FA*IR and binomial FA*IR if only one protected group is present, we see from the results that the colorblind ranking is judged as already fair by binomial FA*IR, but not by multinomial FA*IR. As described in Section 5, the significance adjustment of the multinomial method has variance in its precision. We may therefore observe a slight difference between the analytical α calculated by binomial FA*IR, and the experimental α calculated by multinomial FA*IR. This has no impact on the fair ranking for small p and k. In D1 with $k = 1500$ the small difference in α can lead to slightly different result rankings. Thus we recommend to use the binomial version of FA*IR if only one protected group exists, as it guarantees the best possible significance adjustment (see Zehlike, Sühr, et al. (2020b) for details).

In summary, while the binomial version of FA*IR is computationally preferable, it cannot handle cases where discriminatory patterns manifest differently across protected groups, as we explained in the introduction.

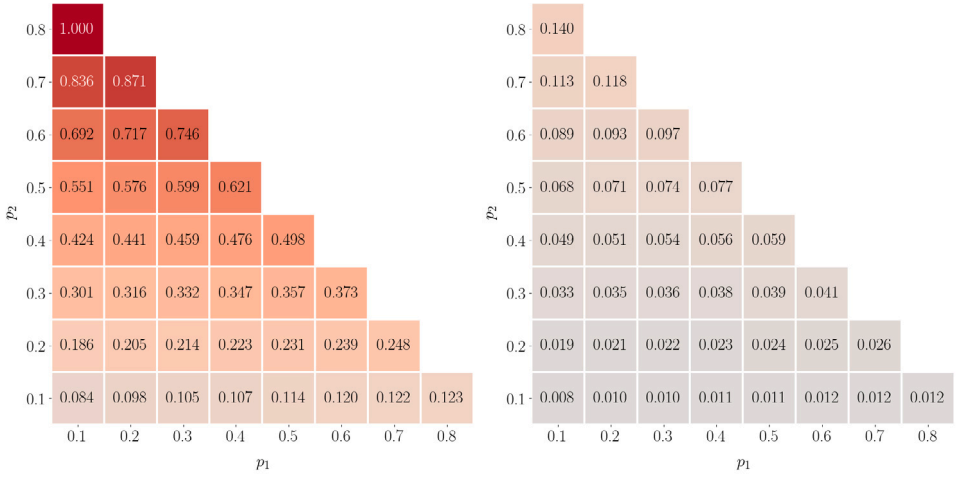
A wide range of possibilities to set p_G. An important advantage of FA*IR is, that it allows to create rankings for user-defined values of p and in particular for values beyond statistical parity, something that cannot be done directly with CFAθ (Zehlike, Hacker, et al. (2020) allows at maximum statistical parity in the result ranking when setting $\theta = 1$, *i.e.*, to the maximum value).

Fig. 6(d) shows results when varying p_1 and p_2 in dataset D2: COMPAS, the protected groups are people of age 25–45 (p_1) and people under 25 years (p_2). In Fig. 6 we see that the non-protected group loses exposure under all settings, as expected. At the same time however NDCG loss is minor and under many settings not increasing beyond a statistical significance level of 0.05% (Fig. 6(d)). Comparing Figs. 6(b) and 6(c) we see that in general, if one protected group scores significantly better in the colorblind ranking (group "25–45" in Fig. 5(c)) than another one (group "younger 25" in Fig. 5(c)), the first may be ranked down to make room for the second. Fig. 6(b) shows that the middle-aged group in the COMPAS dataset loses exposure to the young group unless



(a) Exposure loss for the non-protected group.

(b) Exposure gain/loss for group "age 25 – 45".



(c) Exposure gain for group "age < 25".

(d) NDCG loss.

Fig. 6. Normalized exposure gain/loss and NDCG loss w.r.t. the colorblind ranking in experiment D2 (COMPAS with age as protected category) for different values of p_1 and p_2 ($k = 200, \alpha = 0.1$). p_1 relates to group "age 25–45", p_2 relates to group "age < 25". Blue fields in Fig. 6–6(c) indicate that this group lost exposure w.r.t. the colorblind ranking, red fields indicate exposure gain. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

p_1 is set high enough. However even if p_1 is set low it still ensures that candidates from the middle-aged group show in the ranking, thus always losing less exposure than the non-protected group (compare Fig. 6 and 6(b)). Note that this experiment is an extreme case, where no candidates from group "age < 25" where ranked in the top-200 in $cb|_k$, which explains why this group never loses exposure under any setting for p_1 and p_2 (see Fig. 6(c)).

This experiment shows that FA*IR allows a wide range of positive actions, for instance, granting reintegration programs to people with promising COMPAS ratings, with a preference towards younger offenders. In this case, Fig. 6(d) shows that we can double the proportion of young people in the top-k ranking (e.g. from 0% in the original up to 40% in the fair ranking) without introducing a large drop for the other protected group loss: in the colorblind ranking they received exposure equivalent to $0.4 > p_1 > 0.5$, so setting $p_1 = p_2 = 0.4$ only causes an exposure drop of 5.8%. At the same time the NDCG loss is just 5.6% and never exceeds 15%, even when favoring young people with $p_2 = 0.8$.

8. Conclusions

In this paper we presented the extension of FA*IR to multiple groups, where we guarantee ranked group fairness, without introducing a large utility loss. Especially when groups largely have the same utility score in the top positions (as is the case in the

COMPAS experiments) no ranking utility at all is lost in terms of NDCG or individual fairness. In this case FA*IR only benefits the protected groups without skewing the ranking result. If a protected group already receives advantageous exposure in the colorblind ranking and the ranked group fairness condition is already met via the ranking scores of the candidates, FA*IR preserves this. A protected candidate can only lose exposure due to a protected candidate from another group being ranked up, but not due to a non-protected one. Additionally, the user can control the degree of fairness that is obtained in the result by setting p_G to a value that is appropriate for the situation at hand. This lets them transparently control the trade-off between fairness and utility, instead of having a less intuitive fairness parameter θ that operates on the barycenter of group distributions. By reporting the largest utility loss and rank drop that an individual receives, we explicitly expressed trade-offs from our group fairness criterion to individual fairness as defined by Dwork et al. (2012).

Future work. An important challenge is the algorithmic complexity of calculating the mTree for a particular configuration of p_G and α. Though we already implemented improvements to reduce complexity, calculating an adjusted mTree of length $k = 100$ with six protected groups takes several weeks. Of course, this mTree has to be calculated only once and can then be persistent and shared among users of FA*IR, however when a situation demands a new configuration these calculation times are currently unavoidable. A significant speed-up could be achieved by programming a customized MCDF-function which can store the results of repetitive computation steps. This is however very memory-intensive and the algorithm then needs to run on large computer clusters. Moreover, one could provide a script that fills the MCDF Cache with various configurations of p_G and α. Future work could also investigate other routes of optimizing the creation of mTrees. One field of action could be the parallelization of mTree calculations during significance adjustment. Furthermore, a tailored multinomial CDF function which reuses recurring computation steps might boost the mTree construction significantly. This calculation can continuously run as a separate process on a server which then provides the obtained caches to users who want to compute new mTrees.

One of the main challenges for fair ranking algorithms in general is that there is not yet much empirical evidence that re-ordering items actually helps to overcome the bias in click-probability across groups. Recent research (Sühr, Hilgard, & Lakkaraju, 2020), however, suggests that guarantees for a minimum representation of underrepresented groups yield to higher selection rates in different hiring contexts, but does not mitigate user biases completely. For example, if a user prefers male candidates for a moving assistance task over female candidates, ranking female candidates higher will not mitigate users' biases completely. Thus, a method such as FA*IR may be able to increase the click probability for protected groups by setting the values for p_G higher than the desired click probability. For example, it might be effective to set the minimum proportion for the protected group women in the context of hiring for moving assistance to 60% in order to achieve a click probability of 50% for this group. Nonetheless the study also shows that user biases are complex and hard to mitigate through ranking algorithms alone. For example, users in their study relied heavily on the displayed features, suggesting that showing or hiding certain features may impact user decisions significantly. Further research has to be conducted to study the effect on users of re-ordering items in a ranking and to understand the best means to overcome these strong prejudices against minority groups in certain domains. This is an empirical question that needs to be addressed through user studies; approaches based on simulating clicks using pre-existing user preferences, such as the one used by Abdollahpouri, Mansoury, Burke, Mobasher, and Malthouse (2021), may not uncover the actual interplay between the displayed ranking and latent user preferences.

Additionally, further experimental research using synthetic data could allow us to test with a wider range of differences across groups, larger than the one that real datasets exhibit. This can help us better understand the trade-offs between individual losses and increased group fairness. Our experiments show that, while decreases in NDCG are often negligible, the rank drop for an (unlucky) individual candidate from the non-protected group can still be large. An interesting next step for our method would be to use individual rank drops as input for a second fairness-enhancing method, that could try to optimize for individual fairness, while maintaining the ordering of groups that is required by a given mTree. Finally, robustness tests to measure the sensitivity of the rankings to noise in the qualification/score inputs could be helpful to determine to what extent they may affect our fairness objectives.

Reproducibility. Code and data that can be used to reproduce the experiments on this paper is available: https://github.com/MilkaLichtblau/Multinomial_FA-IR.

Acknowledgments

This research was supported by the Max Planck Institute for Software Systems, the German Research Foundation, Germany and the Catalonia Trade and Investment Agency (ACCIÓ). M.Z. was supported by the MPI and the GRF, Germany. C.C. was partially supported by "la Caixa" Foundation (ID 100010434), under the agreement LCF/PR/PR16/51110009.

Appendix

Algorithm 5: Algorithm MultinomialAlphaAdjustment calculates the corrected significance level α_c for multiple protected groups, such that the mTree $m(\alpha_c, k, p_G)$ has the probability of rejecting a fair ranking α

input : k, the size of the ranking to produce; p_G, the vector of expected proportions of protected elements; α, the desired significance level, ϵ the tolerance for variance in the experimental fail probability calculation.
output : α_{corr} the adjusted significance level, `m_adjusted` the adjusted mTree
// initialize all needed variables
aMin ← 0; aMax ← α ; aMid ← $\frac{(\text{aMin + aMax})}{2}$
m_min ← computeMTree(k,p_G,aMin) m_max ← computeMTree(k,p_G,aMax) m_mid ← computeMTree(k,p_G,aMid)
while *True* **do**
  if *m_mid.getFailProb()* $< \alpha$ **then**
    aMin ← aMid
    m_min ← computeMTree(k,p_G,aMin)
    aMid ← $\frac{\text{aMin+aMax}}{2}$
    m_mid ← computeMTree(k,p_G,aMid)
  end
  if *m_mid.getFailProb()* $> \alpha$ **then**
    aMax ← aMid
    m_max ← computeMTree(k,p_G,aMax)
    aMid ← $\frac{\text{aMin+aMax}}{2}$
    m_mid ← computeMTree(k,p_G,aMid)
  end
  // compute all differences between fail probability and α
  midDiff ← |m_mid.getFailProb() − α|
  maxDiff ← |m_max.getFailProb() − α|
  minDiff ← |m_min.getFailProb() − α|
  // case where midDiff is the smallest difference from desired significance
  if $midDiff \leq \epsilon$ *AND* $midDiff \leq maxDiff$ *AND* $midDiff \leq minDiff$ **then**
    return *aMid, m_mid*
  end
  // case where minDiff is the smallest difference from desired significance
  if $minDiff \leq \epsilon$ *AND* $minDiff \leq maxDiff$ *AND* $minDiff \leq midDiff$ **then**
    return *aMin, m_min*
  end
  // case where maxDiff is the smallest difference from desired significance
  if $maxDiff \leq \epsilon$ *AND* $maxDiff \leq minDiff$ *AND* $maxDiff \leq midDiff$ **then**
    return *aMax, m_max*
  end
end

References

Abdollahpouri, Himan, Mansoury, Masoud, Burke, Robin, Mobasher, Bamshad, & Malthouse, Edward (2021). User-centered evaluation of popularity bias in recommender systems. arXiv preprint arXiv:2103.06364.

Agrawal, Rakesh, Gollapudi, Sreenivas, Halverson, Alan, & Ieong, Samuel (2009). Diversifying search results. In *Proc. of WSDM* (pp. 5–14). ACM.

Altenburger, Kristen, De, Rajlakshmi, Frazier, Kaylyn, Avteniev, Nikolai, & Hamilton, Jim (2017). Are there gender differences in professional self-promotion? an empirical case study of linkedin profiles among recent mba graduates. In *Proceedings of the international AAAI conference on web and social media*, vol. 11.

Angwin, Julia, Larson, Jeff, Mattu, Surya, & Kirchner, Lauren (2016). Machine bias. *ProPublica*.

Barocas, Solon, & Selbst, Andrew D. (2016). Big data's disparate impact. *California Law Review*, *104*, 671.

Beutel, Alex, Chen, Jilin, Doshi, Tulsee, Qian, Hai, Wei, Li, Wu, Yi, et al. (2019). Fairness in recommendation ranking through pairwise comparisons. In *Proc. of KDD* (pp. 2212–2220). ACM.

Biega, Asia J., Gummadi, Krishna P., & Weikum, Gerhard (2018). *Equity of attention: Amortizing individual fairness in rankings* (pp. 405–414). ACM.

Bonchi, Francesco, Hajian, Sara, Mishra, Bud, & Ramazzotti, Daniele (2017). Exposing the probabilistic causal structure of discrimination. *International Journal of Data Science and Analytics*, *3*(1), 1–21.

Burke, Robin, Sonboli, Nasim, & Ordonez-Gauger, Aldo (2018). Balanced neighborhoods for multi-sided fairness in recommendation. In *Conference on fairness, accountability and transparency* (pp. 202–214). PMLR.

Calders, Toon, & Verwer, Sicco (2010). Three naive Bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, *21*(2), 277–292.

Carbonell, Jaime, & Goldstein, Jade (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. of SIGIR* (pp. 335–336). ACM Press.

Castillo, Carlos (2018). Fairness and transparency in ranking. *SIGIR Forum*, *52*.

Celis, L Elisa, Deshpande, Amit, Kathuria, Tarun, & Vishnoi, Nisheeth K (2016). How to be fair and diverse? ArXiv:1610.07183.

Celis, L. Elisa, Straszak, Damian, & Vishnoi, Nisheeth K. (2018). Ranking with fairness constraints. In *Proc. of ICALP, vol. 107*. Schloss Dagstuhl, 28:1–28:15.

Chakraborty, Abhijnan, Patro, Gourab K, Ganguly, Niloy, Gummadi, Krishna P, & Loiseau, Patrick (2019). Equality of voice: Towards fair representation in crowdsourced top-k recommendations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (pp. 129–138).

Channamsetty, Sushma, & Ekstrand, Michael D. (2017). Recommender response to diversity and popularity bias in user profiles (short paper). In *Proc. of florida artificial intelligence research society conference*.
Corbett-Davies, Sam, Pierson, Emma, Feller, Avi, Goel, Sharad, & Huq, Aziz (2017). Algorithmic decision making and the cost of fairness. In *Proc. of KDD* (pp. 797–806). ACM.
Craswell, Nick, Zoeter, Onno, Taylor, Michael J., & Ramsey, Bill (2008). An experimental comparison of click position-bias models. In *Proc. of WSDM* (pp. 87–94). ACM Press.
Dwork, Cynthia, Hardt, Moritz, Pitassi, Toniann, Reingold, Omer, & Zemel, Richard (2012). Fairness through awareness. In *Proc. of ITCS* (pp. 214–226). ACM Press.
Edizel, Bora, Bonchi, Francesco, Hajian, Sara, Panisson, André, & Tassa, Tamir (2020). FaiRecSys: mitigating algorithmic bias in recommender systems. *International Journal of Data Science and Analytics*, *9*(2), 197–213.
Ekstrand, Michael D., & Kluver, Daniel (2021). Exploring author gender in book rating and recommendation. *User Modeling and User-Adapted Interaction*, 1–44.
Ellis, Evelyn, & Watson, Philippa (2012). Eu anti-discrimination law. Oxford University Press.
Feldman, Michael, Friedler, Sorelle A, Moeller, John, Scheidegger, Carlos, & Venkatasubramanian, Suresh (2015). Certifying and removing disparate impact. In *Proc. of KDD* (pp. 259–268). ACM Press.
Friedler, Sorelle A, Scheidegger, Carlos, & Venkatasubramanian, Suresh (2016). On the (im) possibility of fairness. ArXiv:1609.07236.
Friedman, Batya, & Nissenbaum, Helen (1996). Bias in computer systems. *ACM Transactions on Information Systems*, *14*(3), 330–347.
Geyik, Sahin Cem, Ambler, Stuart, & Kenthapadi, Krishnaram (2019). Fairness-aware ranking in search & recommendation systems with application to LinkedIn talent search. In *Proce. of KDD* (pp. 2221–2231). ACM.
Grgić-Hlača, Nina, Zafar, Muhammad Bilal, Gummadi, Krishna P, & Weller, Adrian (2017). On fairness, diversity and randomness in algorithmic decision making. arXiv preprint arXiv:1706.10208.
Hajian, Sara, Bonchi, Francesco, & Castillo, Carlos (2016). Algorithmic bias: From discrimination discovery to fairness-aware data mining. In *KDD tutorials*. ACM.
Hajian, Sara, & Domingo-Ferrer, Josep (2013). A methodology for direct and indirect discrimination prevention in data mining. *IEEE Transactions on Knowledge and Data Engineering*, *25*(7).
Hajian, Sara, Domingo-Ferrer, Josep, & Farràs, Oriol (2014). Generalization-based privacy preservation and discrimination prevention in data publishing and mining. *Data Mining and Knowledge Discovery*, *28*(5–6), 1158–1188.
Hardt, Moritz, Price, Eric, & Srebro, Nati (2016). Equality of opportunity in supervised learning. In *Proc. of NIPS* (pp. 3315–3323). Curran Associates, Inc..
Heidari, Hoda, Loi, Michele, Gummadi, Krishna P, & Krause, Andreas (2019). A moral framework for understanding fair ML through economic models of equality of opportunity. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (pp. 181–190).
International Labor Organization (2019). *Promoting employment opportunities for people with disabilities*: *Technical report*.
Jabbari, Shahin, Joseph, Matthew, Kearns, Michael, Morgenstern, Jamie, & Roth, Aaron (2016). Fair learning in Markovian environments. In *Proc. of FATML*.
Jannach, Dietmar, Lerche, Lukas, Kamehkhosh, Iman, & Jugovac, Michael (2015). What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction*, *25*(5), 427–491.
Järvelin, Kalervo, & Kekäläinen, Jaana (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, *20*(4), 422–446.
Kamiran, Faisal, Calders, Toon, & Pechenizkiy, Mykola (2010). Discrimination aware decision tree learning. In *Proc. of ICDM* (pp. 869–874). IEEE CS.
Kamishima, Toshihiro, Akaho, Shotaro, Asoh, Hideki, & Sakuma, Jun (2012). Fairness-aware classifier with prejudice remover regularizer. In *Proc. of ECML/PKDD* (pp. 35–50). Springer.
Kamishima, Toshihiro, Akaho, Shotaro, Asoh, Hideki, & Sakuma, Jun (2018). Recommendation independence. In *Conference on fairness, accountability and transparency* (pp. 187–201). PMLR.
Kleinberg, Jon, Lakkaraju, Himabindu, Leskovec, Jure, Ludwig, Jens, & Mullainathan, Sendhil (2018). Human decisions and machine predictions. *Quarterly Journal of Economics*, *133*(1), 237–293.
Kuhlman, Caitlin, & Rundensteiner, Elke (2020). Rank aggregation algorithms for fair consensus. *13*, (12), (pp. 2706–2719). VLDB Endowment,
Kuhlman, Caitlin, VanValkenburg, MaryAnn, & Rundensteiner, Elke (2019). Fare: Diagnostics for fair ranking using pairwise error metrics. In *Proc. of the web conference* (pp. 2936–2942). ACM.
Kulshrestha, Juhi, Zafar, Muhammad B., Eslami, Motahhare, Ghosh, Saptarshi, Messias, Johnnatan, & Gummadi, Krishna P. (2017). Quantifying search bias: Investigating sources of bias for political searches in social media. In *Proc. of CSCW*. ACM.
Kunaver, Matevž, & Požrl, Tomaž (2017). Diversity in recommender systems–a survey. *Knowledge-Based Systems*, *123*, 154–162.
Lahoti, Preethi, Gummadi, Krishna P., & Weikum, Gerhard (2019). Operationalizing individual fairness with pairwise fair representations. In *Proc. VLDB endow., vol. 13* (pp. 506–518).
Lerner, Nātān (2003). Group rights and discrimination in international law. *77*, Martinus Nijhoff Publishers.
Lichman, M. (2013). UCI machine learning repository.
McNair, Douglas S. (2018). Preventing disparities: Bayesian and frequentist methods for assessing fairness in machinelearning decision-support models. *New Insights Into Bayesian Inference*, 71.
O'Neil, Cathy (2016). *Weapons of math destruction: how big data increases inequality and threatens democracy*. Crown Publishing Group.
Pedreschi, Dino, Ruggieri, Salvatore, & Turini, Franco (2009a). Integrating induction and deduction for finding evidence of discrimination. In *Proc. of AI and Law* (pp. 157–166). ACM Press.
Pedreschi, Dino, Ruggieri, Salvatore, & Turini, Franco (2009b). Measuring discrimination in socially-sensitive decision records. In *Proc. of SDM* (pp. 581–592). SIAM.
Pedreshi, Dino, Ruggieri, Salvatore, & Turini, Franco (2008). Discrimination-aware data mining. In *Proc. of KDD* (pp. 560–568). ACM Press.
Raghavan, Manish, Barocas, Solon, Kleinberg, Jon, & Levy, Karen (2020). Mitigating bias in algorithmic hiring: Evaluating claims and practices. In *Proc. of FAT** (pp. 469–481). ACM.
Rosenbaum, Howard, & Fichman, Pnina (2019). Algorithmic accountability and digital justice: A critical assessment of technical and sociotechnical approaches. In *Proc. of ASIST*, vol. 56, 1 (pp. (237–244).
Sakai, Tetsuya, & Song, Ruihua (2011). Evaluating diversified search results using per-intent graded relevance. In *Proc. of SIGIR* (pp. 1043–1052). ACM Press.
Shang, Jin, Sun, Mingxuan, & Lam, Nina S. N. (2020). List-wise fairness criterion for point processes. In *Proc. of KDD* (pp. 1948–1958). ACM.
Singh, Ashudeep, & Joachims, Thorsten (2018). Fairness of exposure in rankings. In *Proc. of KDD* (pp. 2219–2228). ACM.
Singh, Ashudeep, & Joachims, Thorsten (2019). Policy learning for fairness in ranking. In *Proc. NeurIPS* (pp. 5427–5437).
Sowell, Thomas (2005). *Affirmative action around the world: an empirical analysis*. Yale University Press.
Steck, Harald (2018). Calibrated recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems* (pp. 154–162).
Sühr, Tom, Hilgard, Sophie, & Lakkaraju, Himabindu (2020). Does fair ranking improve minority outcomes? Understanding the interplay of human and algorithmic biases in online hiring. arXiv preprint arXiv:2012.00423.
Sweeney, Latanya (2013). Discrimination in online ad delivery. *Queue*, *11*(3), 10.
The Pennsylvania State University (2018). The multinomial distribution. https://newonlinecourses.science.psu.edu/stat504/node/40/.
Verge, Tània (2010). Gendering representation in Spain: Opportunities and limits of gender quotas. *Journal of Women, Politics and Policy*, *31*(2), 166–190.
Žliobaitė, Indrė (2015). A survey on measuring indirect discrimination in machine learning. ArXiv:1511.00148 (Pre-Print).
Wightman, Linda F., & Ramsey, Henry (1998). *LSAC national longitudinal bar passage study*. Law School Admission Council.

Yang, Ke, & Stoyanovich, Julia (2016) Measuring fairness in ranked outputs. In *Proc. of FATML*.

Yang, Ke, Stoyanovich, Julia, Asudeh, Abolfazl, Howe, Bill, Jagadish, HV, & Miklau, Gerome (2018). A nutritional label for rankings. In *Proceedings of the 2018 international conference on management of data* (pp. 1773–1776). ACM.

Yao, Sirui, & Huang, Bert (2017). Beyond parity: Fairness objectives for collaborative filtering. arXiv preprint arXiv:1705.08804.

Yeom, Samuel, & Tschantz, Michael Carl (2021). Avoiding Disparity Amplification under Different Worldviews. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, (pp. 273–283).

Zafar, Muhammad Bilal, Valera, Isabel, Gomez Rodriguez, Manuel, & Gummadi, Krishna P (2017). Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proc. of the web conference* (pp. 1171–1180). IW3C2.

Zafar, Muhammad Bilal, Valera, Isabel, Rodriguez, Manuel Gomez, & Gummadi, Krishna P (2015). Fairness constraints: A mechanism for fair classification. ArXiv:1507.05259.

Zehlike, Meike, Bonchi, Francesco, Castillo, Carlos, Hajian, Sara, Megahed, Mohamed, & Baeza-Yates, Ricardo (2017). Fa*ir: A fair top-k ranking algorithm. In *Proc. of CIKM* (pp. 1569–1578). ACM.

Zehlike, Meike, & Castillo, Carlos (2020). Reducing disparate exposure in ranking: A learning to rank approach. In *Proc. of the web conference* (pp. 2849–2855).

Zehlike, Meike, Hacker, Philipp, & Wiedemann, Emil (2020). Matching code and law: achieving algorithmic fairness with optimal transport. *Data Mining and Knowledge Discovery*, *34*(1), 163–200.

Zehlike, Meike, Sühr, Tom, & Castillo, Carlos (2020b). A note on the significance adjustment for FA*IR with two protected groups. arXiv preprint arXiv:2012.12795.

Zehlike, Meike, Sühr, Tom, Castillo, Carlos, & Kitanovski, Ivan (2020). FairSearch: A tool for fairness in ranked search results. In *Companion Proceedings of the Web Conference 2020* (pp. 172–175).

Zemel, Rich, Wu, Yu, Swersky, Kevin, Pitassi, Toni, & Dwork, Cynthia (2013). Learning fair representations. In *Proc. of ICML* (pp. 325–333).

Zhu, Ziwei, Hu, Xia, & Caverlee, James (2018). Fairness-aware tensor-based recommendation. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 1153–1162).

Žliobaite, Indre, Kamiran, Faisal, & Calders, Toon (2011). Handling conditional discrimination. In *Proc. of ICDM* (pp. 992–1001). IEEE CS.