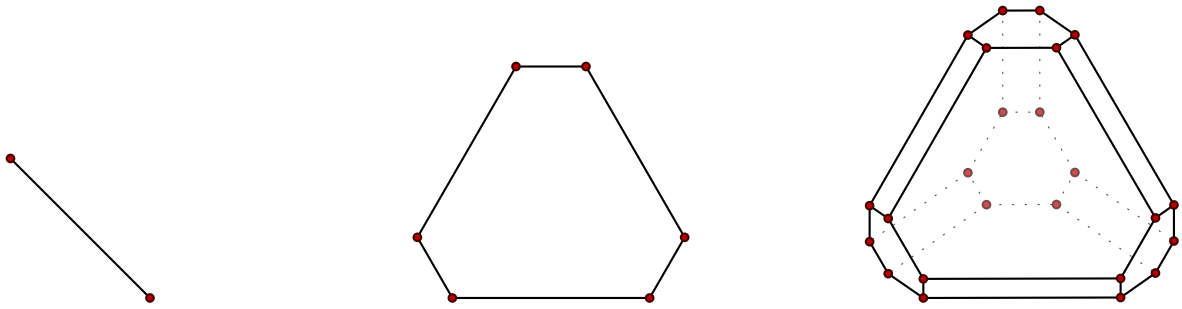
Introducing the Expohedron for Efficient Pareto-optimal Fairness-Utility Amortizations in Repeated Rankings

Till Kletti
till.kletti@naverlabs.com
Naver Labs Europe
France

Jean-Michel Renders
jean-michel.renders@naverlabs.com
Naver Labs Europe
France

Patrick Loiseau
patrick.loiseau@inria.fr
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG
France



(a) $n = 2$. A line segment in $\mathbb{R}^2$ **(b) $n = 3$. A polygon in $\mathbb{R}^3$** **(c) $n = 4$. A polyhedron in $\mathbb{R}^4$**

Figure 1: Examples of the DCG expohedron for $n \in \{2, 3, 4\}$ items. The vertices are the DCG exposures $\left(\frac{1}{\log_2(2)}, \ldots, \frac{1}{\log_2(n+1)}\right)$ under application of the symmetric group $\mathcal{S}_n$. The expohedron is the convex hull of these vertices. Expohedra live in hyperplanes of dimension $n - 1$.

ABSTRACT

We consider the problem of computing a sequence of rankings that maximizes consumer-side utility while minimizing producer-side individual unfairness of exposure. While prior work has addressed this problem using linear or quadratic programs on bistochastic matrices, such approaches, relying on Birkhoff-von Neumann (BvN) decompositions, are too slow to be implemented at large scale.

In this paper we introduce a geometrical object, a polytope that we call *expohedron*, whose points represent all achievable exposures of items for a Position Based Model (PBM). We exhibit some of its properties and lay out a Carathéodory decomposition algorithm with complexity $O(n^2 \log(n))$ able to express any point inside the expohedron as a convex sum of at most n vertices, where n is the number of items to rank. Such a decomposition makes it possible to express any feasible target exposure as a distribution over at most n rankings. Furthermore we show that we can use this polytope to recover the whole Pareto frontier of the multi-objective fairness-utility optimization problem, using a simple geometrical procedure with complexity $O(n^2 \log(n))$. Our approach compares favorably to linear or quadratic programming baselines in terms of algorithmic complexity and empirical runtime and is applicable to any merit that is a non-decreasing function of item relevance. Furthermore our solution can be expressed as a distribution over only n permutations, instead of the $(n-1)^2 + 1$ achieved with BvN decompositions. We perform experiments on synthetic and real-world datasets, confirming our theoretical results.

WSDM '22, February 21–25, 2022, Tempe, AZ, USA

ACM ISBN 978-1-4503-9132-0/22/02…$15.00
https://doi.org/10.1145/3488560.3498490

CCS CONCEPTS

• **Mathematics of computing** → *Permutations and combinations*; • **Information systems** → **Probabilistic retrieval models**; **Content ranking**.

KEYWORDS

ranking, fairness, amortization, pareto-optimal, muli-objective optimization, Carathéodory, expohedron, GLS, balanced words

ACM Reference Format:
Till Kletti, Jean-Michel Renders, and Patrick Loiseau. 2022. Introducing the Expohedron for Efficient Pareto-optimal Fairness-Utility Amortizations in Repeated Rankings. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3488560.3498490

1 INTRODUCTION

Ranking systems are nowadays widely deployed in the world wide web and make it possible for users of search engines to quickly retrieve items that are relevant with respect to their query. Those items can be websites, movies, songs, research papers and many other things. With web search and recommendation having an ever increasing influence on people's behavior and determining which opportunities people are given, the questions whether ranking systems are *fair* and how to make them fair, has gathered much attention in recent years [3, 23, 30].

A typical ranking system estimates a relevance value for each item and orders the items by decreasing relevance values; this is

called the Probability Ranking Principle (PRP) and provides under some assumptions the highest expected utility for the users [27]. However, it has been noted that PRP rankings can be very unfair to item producers [3, 30] insofar as their exposure is not proportional to their merit. For instance when all items are almost equally relevant, the last item will receive much less exposure than the first item, even though their merits are almost equal. To avoid this unfairness, a common approach is to deliver different rankings each time a user issues the query, in such a way that the average exposure given to each item is proportional to its merit. This is called *amortization* [3].

In recent work, Singh and Joachims give a method to compute a distribution over rankings that maximizes utility while satisfying a fairness constraint [30]. They express a distribution over rankings as a bistochastic matrix whose elements represent the proportions with which an item should be at a certain rank. They formulate the utility objective as a linear function of a bistochastic matrix and the fairness objective as a linear constraint on the bistochastic matrix. The optimal bistochastic matrix is then decomposed as a distribution over permutations using the Birkhoff-von Neumann (BvN) algorithm [9]. A similar approach has been used in follow-up work [32, 33, 39]. However it has been noted that such an approach does not scale well [11] as it involves a linear program (LP) over n^2 variables, where n is the number of items to rank. Using state-of-the-art LP solvers adapted to our problem, the total complexity of such an approach is then $O(n^5)$.

In this paper we lay out a method that computes the whole Pareto-frontier of the fairness-utility problem, in less time than it takes an LP to compute just one endpoint of the frontier, both in terms of algorithmic complexity and empirical runtime. Furthermore, our method is applicable to any reasonable notion of merit, requiring only that more relevant items have greater or equal merit than less relevant items.

Our solution strongly relies on the introduction of a geometrical object, a polytope that we call *expohedron*, as it generalizes the well-known permutohedron. While the permutohedron is defined as the convex hull of all permutations of the vector $(1, \dots, n) \in \mathbb{R}^n$, the $\boldsymbol{\gamma}$-expohedron is defined as the convex hull of all permutations of an arbitrary vector $\boldsymbol{\gamma} \in \mathbb{R}^n$. Figure 1 illustrates some expohedra for $n \in \{2, 3, 4\}$. We are not the first to study this kind of polytope, also called *orbit polytope* [21], but to the best of our knowledge, we are the first to apply it to a fair ranking problem. When we set $\boldsymbol{\gamma}$ to be the vector of exposures a.k.a. examination parameters of a Position Based Model (PBM), each vertex of the $\boldsymbol{\gamma}$-expohedron represents the exposures allocated to the items by a certain ranking. A vector obtained by a convex combination of N vertices represents the average exposure obtained by the items when delivering each of the N rankings with proportions given by the convex coefficients.

Our main technical contributions are:

(1) We propose an efficient algorithm that takes any target exposure inside the expohedron as input and finds a distribution over n rankings whose expected value is equal to the target exposure, where n is the number of items to rank. Such an algorithm is called a *Carathéodory decomposition* [4, 20]. Our algorithm has complexity $O(n^2 \log(n))$ and is to the best of our knowledge the first Carathéodory decomposition algorithm in the expohedron.
(2) We propose an efficient algorithm that finds all Pareto-optimal exposure vectors in the expohedron, given a linear utility objective and a quadratic fairness objective as in [7], relying only on simple geometrical constructions. The result of this algorithm is a set of exposure vectors, each of which can be expressed as the expectation of a distribution using contribution (1). This algorithm has complexity $O(n^2 \log(n))$.
(3) As an alternative to randomly *sampling* rankings from a distribution, we propose to use a deterministic scheduling method based on balanced words [36]. Such an approach, as we will argue, better approximates the actual proportions in the distribution, and thus decreases the variance around the expected value when the rankings are actually delivered.
(4) We perform experiments on synthetic data and on the TREC2020 Fairness Track [2] and MSLR [25] datasets to compare our methods to several baselines in terms of Pareto-optimality and efficiency.

2 RELATED WORK

Fairness in ranking. There is a large body of literature that studies problems of fairness in AI systems. Several applications are studied, such as classification [10] or regression [19]. In the context of ranking, there are pre-processing methods [28, 43], whose aim is to "de-bias" the data before it is processed by a ranking algorithm; in-processing methods [43], that aim to modify the ranking algorithm itself; and post-processing methods, which re-order the items only in the end [3, 30, 42]. Our work is a post-processing approach.

Fairness can be enforced on the side of the ranking recipients [8] or on the side of the item producers [3, 30, 42]. We consider fairness on the item producer side.

Fairness of exposure can be enforced at the level of groups [30] with the idea that each group of items should receive a certain amount of exposure, or at the level of the individual with the idea that every item [7] or every item producer [3] should receive a certain amount of exposure. This is called *individual fairness* [3]. In this paper we use individual fairness at the level of the items, i.e., we ensure that each item gets a certain target exposure.

There are many published definitions of fairness. Some *group fairness* definitions [42] look at the items in the top-k ranks and enforce that each group is present with equal proportions for every k. Other approaches called *fairness of exposure* [3, 7, 18, 22, 30, 31, 33, 38–40], like our own approach, seek to adjust the exposure of the items, to a certain range of admissible or optimal values. This is the case with a Learning to Rank approach by Oosterhuis [22] that applies gradient-descent to find the optimal scores of the items when the ranking policy is a Plackett-Luce distribution.

Approaches using fairness of exposure depend on the exposure model that is used. In a Position Based Model (PBM), the exposure of an item depends only on its rank [6]. With other models such as Dynamic Bayesian Network (DBN) [5, 6], the exposure of an item also depends on higher ranked items. Our approach is applicable to PBMs.

Optimizing bistochastic matrices. Singh and Joachims [30] formulate a ranking policy as a bistochastic matrix whose elements are the probabilities for an item to end up at a certain rank. When a PBM is used, it is possible to formulate fairness as a linear constraint

and utility as a linear objective. The problem is solved using a linear program (LP) and an optimal bistochastic matrix is found. It is then possible to decompose the bistochastic matrix as a convex sum of permutation matrices. This is called a Birkhoff-von Neumann (BvN) decomposition and expresses the bistochastic matrix as an expected value of a distribution over permutations. The ranking policy then consists in sampling from this distribution. This approach has later been adopted in subsequent papers [32, 33, 39]. Such an approach finds only one point of the Pareto-frontier: the point with minimal unfairness, whereas our approach takes less time to find the whole frontier. This is achieved by working in the expohedron instead of the Birkhoff polytope (i.e., the set of bistochastic matrices).

Controller. Another more heuristic approach is to use controllers. A controller generally starts by delivering a PRP ranking and computes at every time-step the exposure difference w.r.t. a certain target value. This error term is added to the relevance scores and for the next ranking the items are ordered by this new score. In other words, a controller greedily corrects the empirical unfairness at every time-step by giving adequate bonuses to disadvantaged items. This can be done in a static setting [34] or within a Learning to Rank framework [18].

3 MODEL AND PROBLEM STATEMENT

Setting. We suppose that we are given a query q to which is associated a set of n items indexed by i. In our setting, all queries are treated independently. Therefore in the remainder of this paper, we omit indexing with q. We further suppose that we are given a vector of relevance values $\boldsymbol{\rho} \in [0,1]^n$ that expresses how much each item is relevant to the query. We assume that the query is repeated by the users many times, giving us sufficiently many opportunities to execute our amortization.

Ranking. In this paper we denote $\mathcal{S}_n$ the set of all permutation matrices of size n. We define a ranking as a permutation matrix $\pi \in \mathcal{S}_n$ such that $\pi_{ik} = 1$ if and only if the item i is at rank k.

PBM exposure models. We work with a Position Based Model (PBM) [6], represented by a vector $\boldsymbol{\gamma} \in \mathbb{R}_+^n$ whose k^{th} component is the exposure associated to rank k. We assume w.l.o.g.[1] that the exposure decreases with increasing rank. The exposure is a measure of how much attention the users dedicate to a certain rank.

Given a ranking $\pi \in \mathcal{S}_n$, the exposure associated to the items is given by the vector $\boldsymbol{\mathcal{E}}(\pi) = \pi\boldsymbol{\gamma}$, where the i^{th} element of $\boldsymbol{\mathcal{E}}(\pi)$ is the exposure allocated to item i.

Given a distribution $\mathcal{D}$ of N rankings $\pi_1, \dots, \pi_N$ delivered with proportions $p_1, \dots, p_N$, the expected exposure associated to the items and denoted $\boldsymbol{\mathcal{E}}(\mathcal{D})$ is given by

$$\boldsymbol{\mathcal{E}}(\mathcal{D}) = \sum_{i=1}^{N} p_i \boldsymbol{\mathcal{E}}(\pi_i) = \sum_{i=1}^{N} p_i \pi_i \boldsymbol{\gamma}. \tag{1}$$

Utility. Given a ranking $\pi \in \mathcal{S}_n$, the utility is the scalar product of the relevance vector with the exposures associated to the items:

$$U(\pi) := \boldsymbol{\rho}^\top \boldsymbol{\mathcal{E}}(\pi) = \boldsymbol{\rho}^\top \pi \boldsymbol{\gamma}. \tag{2}$$

[1] It is w.l.o.g., because we can just rename the ranks to make it true.

When one chooses the k^{th} element of $\boldsymbol{\gamma}$ to be $\gamma_k = 1/\log_2(k+1)$ for all $k \in \{1, \dots, n\}$, then $U(\pi)$ is the Discounted Cumulative Gain (DCG) metric [14] of ranking π. When one chooses $\gamma_k = (1-p)p^{k-1}$ for some $p \in (0,1)$, then U is the Rank Biased Precision (RBP) metric [17] of ranking π. Given a distribution $\mathcal{D}$ of N rankings $\pi_1, \dots, \pi_N$ delivered with proportions $p_1, \dots, p_N$, the average utility is

$$U(\mathcal{D}) = \sum_{i=1}^{N} p_i U(\pi) = \sum_{i=1}^{N} p_i \boldsymbol{\rho}^\top \pi_i \boldsymbol{\gamma} = \boldsymbol{\rho}^\top \boldsymbol{\mathcal{E}}(\mathcal{D}). \tag{3}$$

(Un)fairness. Since there exist multiple definitions of fairness, in this paper we assume that a decision-maker has decided which exposures are fair by determining a *target exposure* for each item, for instance as in [7] where the target exposure is defined as an affine function of the relevance vector $\boldsymbol{\rho}$ (some kind of meritocratic fairness). This target is to be understood as the amount of exposure an item deserves each time the query is made. If this target exposure is reached then the system is fair by definition. In order to define how unfair a system is we need a function that measures how far we are from the target exposure. In this paper we proceed as in [7] in adopting the standard euclidean norm. Formally, if we denote $\boldsymbol{\mathcal{E}}^*$ the target exposure and $\boldsymbol{\mathcal{E}}(\mathcal{D})$ the actual expected exposure allocated to the items, then the unfairness is

$$F(\mathcal{D}, \boldsymbol{\mathcal{E}}^*) := \left\| \boldsymbol{\mathcal{E}}(\mathcal{D}) - \boldsymbol{\mathcal{E}}^* \right\|_2. \tag{4}$$

While in [30], fairness is defined as a condition that is either satisfied or not, we consider here a way to quantify it. This will allow us to trade off utility with fairness.

Objective. Note that both U anf F depend on $\mathcal{D}$ only through its expectation so that we can write $U(\boldsymbol{\mathcal{E}}(\mathcal{D}))$ and $F(\boldsymbol{\mathcal{E}}(\mathcal{D}), \boldsymbol{\mathcal{E}}^*)$. Our goal is to solve a Multi-Objective Optimisation (MOO) problem with two objectives, the maximization of utility and the minimization of unfairness after the delivery of a large number of rankings:

$$\max_{\mathcal{D}} U(\boldsymbol{\mathcal{E}}(\mathcal{D})), \ \min_{\mathcal{D}} F(\boldsymbol{\mathcal{E}}(\mathcal{D}), \boldsymbol{\mathcal{E}}^*). \tag{5}$$

In principle one would expect the optimization variable to be a sequence of rankings. However we decompose the problem into 3 distinct sub-steps.

(1) Find all Pareto-optimal expected exposure vectors $\boldsymbol{\mathcal{E}}$. We solve this problem in Section 5.
(2) Given an expected exposure vectors $\boldsymbol{\mathcal{E}}$, find a probability distribution $\mathcal{D}$ over n rankings, whose expectation $\boldsymbol{\mathcal{E}}(\mathcal{D})$ is equal to $\boldsymbol{\mathcal{E}}$. We solve this problem in Section 4.2.
(3) Deliver the rankings of the support of $\mathcal{D}$ on a finite number of samples, with proportions as close as possible to the probabilities in $\mathcal{D}$. We solve this problem in Section 6.

This procedure is analogous to the one used by Singh and Joachims [30]. In their paper Step (1) is done using linear programming, but only an endpoint of the Pareto-frontier is determined and it is empirically found to be slower than our method. Step (2) is done using a BvN decomposition algorithm [9], but the decomposition is done over at most $(n-1)^2 + 1$ rankings, whereas we use at most n and are much quicker doing so. Step (3) is done using sampling, which does not very well approximate the target proportions at low time horizons.

4 THE EXPOHEDRON AND A CARATHÉODORY DECOMPOSITION

The permutation simplex. A naive way of representing distributions over permutations is to consider the space $\mathbb{R}^{n!}$, with each basis vector representing one permutation in $\mathcal{S}_n$. Every point in the $\mathbb{R}^{n!}$-simplex then corresponds to exactly one distribution over permutations, given by its decomposition into basis vectors. Furthermore the $\mathbb{R}^{n!}$-simplex is the convex hull of all permutations as represented in the space $\mathbb{R}^{n!}$. In practice this approach is not feasible, because of the high dimensionality of the space $\mathbb{R}^{n!}$.

The Birkhoff polytope. In the case of PBM exposure models, we can work in a smaller, more tractable space: the space of bistochastic matrices a.k.a. Birkhoff polytope [30]. Indeed in order to know the expected exposure an item gets from a distribution over rankings, it suffices to know the marginal probabilities to end up at each rank. This reduces the dimensionality of the space to n^2. The Birkhoff polytope is also the convex hull of all permutations, represented by permutation matrices. There exists an algorithm, called Birkhoff-von Neumann (BvN) decomposition [9], that can express any bistochastic matrix of size $n \times n$ as a convex sum of at most $(n-1)^2 + 1$ permutation matrices.

The expohedron. It is possible to reduce even further the dimensionality of the space we work in. Indeed what interests us in the end is the exposure that the items get, not with which frequency they are at certain ranks. For a PBM with exposure vector $\boldsymbol{\gamma}$, the $\boldsymbol{\gamma}$-expohedron is formally defined as

$$\Pi(\boldsymbol{\gamma}) \coloneqq \mathrm{Conv}\left(\pi\boldsymbol{\gamma} \mid \pi \in \mathcal{S}_n\right). \tag{6}$$

It is the convex hull of exposure vectors achievable with all possible rankings.

Theorem 1 (Carathéodory [4, 20])**.** *Let $\Pi \in \mathbb{R}^n$ be the convex hull of a finite number of points $\mathbf{v}_1, \ldots, \mathbf{v}_N$. Then any point $\mathbf{x} \in \Pi$ can be expressed as the convex sum of at most $n+1$ points $\mathbf{v}_i$.*

Carathéodory's theorem tells us that it is possible to express any point in the expohedron as a convex sum of at most $n+1$ vertices. Such a convex sum is called *Carathéodory decomposition.* In our case the sum can be made over at most n vertices, because the expohedron lives within a hyperplane of $\mathbb{R}^n$. Indeed by permuting the elements of $\boldsymbol{\gamma}$, the sum of the elements of $\pi\boldsymbol{\gamma}$ remains unchanged. Therefore the vector $\mathbf{1}$ is orthogonal to the expohedron.

A Carathéodory decomposition can express any point $\boldsymbol{\mathcal{E}}$ inside the expohedron as the expected value $\boldsymbol{\mathcal{E}}(\mathcal{D})$ of a distribution $\mathcal{D}$ over at most n rankings.

4.1 Properties of the expohedron

Majorization. The concept of majorization [16] gives us an easy-to-check criterion for verifying if a point is inside the expohedron. We say that a vector $\mathbf{a} \in \mathbb{R}^n$ is *majorized* by a vector $\mathbf{b} \in \mathbb{R}^n$ and we write $\mathbf{a} \prec \mathbf{b}$ if and only if

$$\forall k \in \{1, \ldots, n\},\ \sum_{i=1}^{k} \mathbf{b}_i^{\uparrow} \leq \sum_{i=1}^{k} \mathbf{a}_i^{\uparrow}, \quad \sum_{i=1}^{n} \mathbf{b}_i^{\uparrow} = \sum_{i=1}^{n} \mathbf{a}_i^{\uparrow}, \tag{7}$$

where $\mathbf{a}^{\uparrow}$ and $\mathbf{b}^{\uparrow}$ are the vectors $\mathbf{a}$ and $\mathbf{b}$ with elements ordered from the smallest to the greatest.

Then one can show that a point $\mathbf{x}$ is inside the $\boldsymbol{\gamma}$-expohedron if and only if it is majorized by $\boldsymbol{\gamma}$ [16, 26]. Formally

$$\mathbf{x} \in \Pi(\boldsymbol{\gamma}) \iff \mathbf{x} \prec \boldsymbol{\gamma}. \tag{8}$$

Order-preserving zone. We introduce the concept of *order-preserving zones* or *zones* as $n!$ subsets of $\mathbb{R}^n$, each corresponding to a permutation. We denote by $Z(\pi)$ the zone corresponding to the set of vectors $\mathbf{x}$ such that the elements of $\pi\mathbf{x}$ are sorted in increasing order. Figures 2 and 3 illustrate the subdivision of an expohedron into 6 zones for $n = 3$ documents. If a point $\mathbf{x}$ has ties, then we consider the point to be a member of all zones corresponding to the possible orderings.

Face characterization. Recall that a face of a polytope is a polytope on its boundary. For instance vertices, edges and facets of a dice, are all faces of the dice. We give an easy-to-check criterion for identifying the lowest-dimensional face in which a point of the expohedron lies. A face of the expohedron is characterized by a zone and a subset of $\{1, \ldots, n\}$, that we call *splits*.

Specifically, the face characterized by zone $Z(\pi)$ and by the splits S, is composed of all the points $\mathbf{x} \in \Pi(\boldsymbol{\gamma})$ satisfying the condition

$$\forall i \in S,\ \sum_{k=1}^{i} (\pi\mathbf{x})_k = \sum_{k=1}^{i} \boldsymbol{\gamma}_k^{\uparrow}, \tag{9}$$

where $\boldsymbol{\gamma}^{\uparrow}$ is the vector $\boldsymbol{\gamma}$ with elements ordered from the smallest to the greatest. The dimension of a face is given by $\dim(F) = n - |S|$. For a vertex, we have $S = \{1, \ldots, n\}$ and its dimension is 0. For the whole polytope, we have $S = \{n\}$ and its dimension is $n-1$. From the majorization property it follows that all faces of the expohedron have n in their splits.

With this property it becomes easy to determine the faces in which a point $\mathbf{x}$ of the expohedron lies. It suffices to set π such that $\pi\mathbf{x}$ is sorted in increasing order and to check for which i (9) holds.

4.2 Carathéodory decomposition

The expohedron is a convex hull of dimension $n-1$, so from Carathéodory's theorem [4, 20] we know that any of its points can be expressed as a convex combination of at most n of its vertices. This is already a significant improvement over the approach using BvN decomposition, as BvN decompositions use at most $(n-1)^2+1$ permutation matrices [15].

There exist already some published Carathéodory decomposition algorithms for permutohedra [13, 41], but they cannot naturally be extended to expohedra. Expohedra are not always zonotopes, which is a key property used in [13], and they are generally not defined with $\boldsymbol{\gamma} = (1, \ldots, n)$, which is a key property used in [41].

In this section we propose to adapt a generic Carathéodory algorithm, the GLS method named after Grötschel, Lovász and Schrijver [12]. Its main idea is illustrated in Figure 2.

The GLS method for decomposing a point $\mathbf{x}$ consists in choosing an arbitrary vertex $\mathbf{v}_1$ and drawing a half-line starting from $\mathbf{v}_1$ and passing through $\mathbf{x}$. This half-line intersects the polytope's boundary at a point $\mathbf{p}_2$ lying on a face of dimension $n-2$. It is easy to see that $\mathbf{x}$ can be expressed as a convex sum of $\mathbf{v}_1$ (a vertex) and $\mathbf{p}_2$ (not necessarily a vertex). The point $\mathbf{p}_2$ is then decomposed using the same method: we choose an arbitrary vertex $\mathbf{v}_2$ from the face of $\mathbf{p}_2$ and draw a half-line from $\mathbf{v}_2$ through $\mathbf{p}_2$ intersecting the polytope's

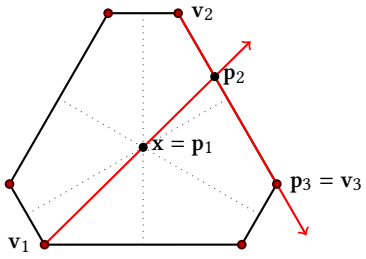


Figure 2: GLS procedure, inspired from an illustration in [13]. The barycenter $\mathbf{x}$ of the expohedron is decomposed into a convex sum of vertices $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$. The dotted grey lines materialize the subdivision of the expohedron into order-preserving zones.

boundary at a point $\mathbf{p}_3$ lying on a face of dimension $n-3$ and we express $\mathbf{p}_2$ as a convex sum of $\mathbf{v}_2$ and $\mathbf{p}_3$. Repeating this procedure recursively, there will come a point at which the intersection $\mathbf{p}_n$ is itself a vertex of the polytope and we are finished.

This procedure has two delicate operations that need to be adapted to the particular polytope on which it is applied:

(1) We need a method to find the intersection of a half-line with the polytope's boundary;
(2) For a point on a face, we need to find a vertex on the same face.

To find a vertex on the same face as a point $\mathbf{x}$, it suffices to permute the elements of $\boldsymbol{\gamma}$ such that its result to `argsort` is the same as for $\mathbf{x}$. This operation has complexity $O(n \log(n))$, because it is as complex as a sorting operation, and can be expressed in python[2] as $\boldsymbol{\gamma}[\texttt{argsort}(\texttt{argsort}(-\mathbf{x}))]$.

For Step (1), one can use the majorization criterion to make a bisection search on the half-line. Such a bisection search has complexity $O(n \log(n))$, because the sort operation to check the majorization needs to be done $O(1)$ times.

Algorithm 1 The GLS method in the expohedron.

procedure GLS(INPUT: $\mathbf{x} \in \Pi(\boldsymbol{\gamma})$)
  $\mathbf{v}_1 \leftarrow \boldsymbol{\gamma}[\texttt{argsort}(\texttt{argsort}(-\mathbf{x}))]$ ▷ Choose an initial vertex
  $\alpha_1 \leftarrow 1$ ▷ Set the initial vertex's weight to 1
  $\mathbf{p}_1 \leftarrow \mathbf{x}$
  for $i \in \{1, \dots, n\}$ **do**
    $\mathbf{p}_{i+1} \leftarrow \max_{\lambda \geq 1} \mathbf{v}_i + \lambda(\mathbf{p}_{i+1} - \mathbf{v}_i)$ s.t. $\mathbf{v}_i + \lambda(\mathbf{p}_{i+1} - \mathbf{v}_i) \prec \boldsymbol{\gamma}$
    $\alpha_{i+1} \leftarrow \alpha_i - \frac{\|\mathbf{p}_i - \mathbf{p}_{i+1}\|}{\|\mathbf{p}_i - \mathbf{v}_i\|}\alpha_i$ ▷ Update convex coefficients
    $\alpha_i \leftarrow \frac{\|\mathbf{p}_i - \mathbf{p}_{i+1}\|}{\|\mathbf{p}_i - \mathbf{v}_i\|}\alpha_i$
    $\mathbf{v}_{i+1} \leftarrow \boldsymbol{\gamma}[\texttt{argsort}(\texttt{argsort}(-\mathbf{p}_{i+1}))]$
  end for
end procedure OUTPUT: $\alpha_1, \dots, \alpha_n, \mathbf{v}_1, \dots, \mathbf{v}_n$

Theorem 2. *Algorithm 1 find a Carathéodory decomposition of any point in the expohedron in $O(n^2 \log(n))$ time.*

Indeed the GLS procedure consists in executing at most n times the steps (1) and (2), which have complexity $O(n \log(n))$, so the complexity of the algorithm we propose is $O(n^2 \log(n))$.

[2] Assuming $\boldsymbol{\gamma}$ has decreasing components.

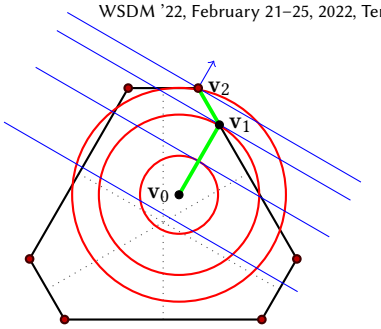


Figure 3: The Pareto curve in a DCG expohedron for $\boldsymbol{\rho} = (0.55, 0.6, 0.65)$ represented as a blue arrow. The point $\mathbf{v}_0$ is the meritocratic target exposure, i.e., $\frac{\|\boldsymbol{\gamma}\|_1}{\|\boldsymbol{\rho}\|_1}\boldsymbol{\rho}$. The red circles are equi-unfairness curves and the blue lines are equi-utility lines. The green line segments form the Pareto-curve in the expohedron. The dotted grey lines materialize the subdivision of the expohedron into order-preserving zones.

5 FINDING THE PARETO CURVE

In this section we show how we can solve the MOO problem (5). We assume that the target exposure $\boldsymbol{\mathcal{E}}^*$ is in the same zone as the PRP ranking. This is a weak assumption, because it is another way of saying that items with higher relevance must not get less exposure than items with lower relevance. Furthermore we assume that the target exposure is inside the expohedron, i.e., the target is feasible.

The decision variable $\boldsymbol{\mathcal{E}}$ also needs to be inside the expohedron, in order to be the expected value of a distribution over rankings (i.e., a convex sum of vertices). So our constraint can be written as $\boldsymbol{\mathcal{E}} \prec \boldsymbol{\gamma}$. Formally our optimization problem is:

$$\max_{\boldsymbol{\mathcal{E}}} \boldsymbol{\rho}^\top \boldsymbol{\mathcal{E}}, \quad \min_{\boldsymbol{\mathcal{E}}} \left\|\boldsymbol{\mathcal{E}} - \boldsymbol{\mathcal{E}}^*\right\|_2^2, \quad \text{s.t. } \boldsymbol{\mathcal{E}} \prec \boldsymbol{\gamma}. \tag{10}$$

The maximization objective is linear and the minimization objective is quadratic and isotropic. The level sets of the utility objective are hyperplanes with orthogonal vector $\boldsymbol{\rho}$ and the level sets of the fairness objective are hyperspheres centered at $\boldsymbol{\mathcal{E}}^*$.

It is now easy to see that amongst all points on an equi-utility hyperplane, the point that minimizes unfairness is the one at which an equi-unfairness hypersphere is tangent, see Figure 3. So to find the Pareto curve, we just need to start from $\mathbf{v}_0 = \boldsymbol{\mathcal{E}}^*$ and go in the direction of $\boldsymbol{\rho}$ until we intersect the border of the expohedron. This intersection can be computed analytically using (12), because we stay in the same zone by moving in the direction of $\boldsymbol{\rho}$. Indeed the intersection of a half-line $\mathbf{v} + \lambda\boldsymbol{\rho}$ with the expohedron must satisfy

$$\lambda = \arg\max_{\mu \geq 0} \mathbf{v} + \mu\boldsymbol{\rho} \quad \text{s.t.} \quad \mathbf{v} + \mu\boldsymbol{\rho} \prec \boldsymbol{\gamma}. \tag{11}$$

The elements of $\mathbf{v} + \lambda\boldsymbol{\rho}$ stay in the same order for $\lambda \geq 0$, because $\mathbf{v}$ and $\boldsymbol{\rho}$ are in the same zone. It follows after some algebraic manipulations that

$$\lambda = \min_k \left\{ \frac{G_k - V_k}{D_k} | D_k < 0 \right\}, \tag{12}$$

where $G_k = \sum_{i=1}^k \boldsymbol{\gamma}_i^{\uparrow}$, $V_k = \sum_{i=1}^k \mathbf{v}_i^{\uparrow}$, $D_k = \sum_{i=1}^k (\mathbf{x} - \mathbf{v})_i^{\uparrow}$, and the arrow $\uparrow$ indicates a vector sorted from smallest to greatest element.

Once we are at the intersection, we can apply a similar reasoning to get the next segment of the Pareto curve. The intersection of a hypersphere with the affine subspace containing the current face is another hypersphere of lower dimension. Similarly the intersection of a utility hyperplane with the affine subspace containing the

current face is itself an affine subspace of lower dimension. Thus we need to go in the direction of the projection of $\boldsymbol{\rho}$ on the affine subspace containing the current face, until we meet the border of the expohedron again in a face of lower dimension. In the end we are guaranteed to end up at a point maximizing utility, which constitutes the endpoint of the Pareto curve.

Algorithm 2 Pareto set identification. We assume w.l.o.g. that the zone $Z(\pi)$ of $\boldsymbol{\rho}, \mathbf{v}^{(0)}$ is the one where the values are ordered from smallest to greatest. This corresponds to making a change of basis such that π becomes the identity matrix. We denote $\boldsymbol{\gamma}^{\uparrow}$ the vector $\boldsymbol{\gamma}$ ordered from smallest to greatest element.

procedure Pareto(INPUT: $\boldsymbol{\gamma}$, target exposure $\mathbf{v}^{(0)}$, $\boldsymbol{\rho}$)
  $G_k \leftarrow \sum_{i=1}^{k} \boldsymbol{\gamma}_i^{\uparrow}$
  $V_k \leftarrow \sum_{i=1}^{k} \mathbf{v}_i^{(0)}$
  $\mathcal{I}_0 \leftarrow \{n\}$ ▷ Initilize the set of splits
  $\boldsymbol{\rho}^{(0)} \leftarrow \boldsymbol{\rho} - (\boldsymbol{\rho}^{\top}\mathbf{1})\mathbf{1}/n$ ▷ Project $\boldsymbol{\rho}$ on the hyperplane with normal vector $\mathbf{1}$
  $l \leftarrow 0$
  while $\boldsymbol{\rho}^{\top}\mathbf{v}^{(l)} < \boldsymbol{\rho}^{\top}\boldsymbol{\gamma}^{\uparrow}$ **do** ▷ While utility is not maximal
    $D_k \leftarrow \sum_{i=1}^{k} \boldsymbol{\rho}_i^{(l)}$
    $\lambda_l \leftarrow \min_k \left\{ \frac{G_k - V_k}{D_k} | D_k < 0 \right\}$
    $\mathbf{v}^{(l+1)} \leftarrow \mathbf{v}^{(l)} + \lambda_l \boldsymbol{\rho}^{(l)}$ ▷ Compute the intersection
    $V_k \leftarrow \sum_{i=1}^{k} \mathbf{v}_i^{(l+1)}$
    $\mathcal{I}_{l+1} = \{i_1, \ldots, i_{l+1}\} \leftarrow \texttt{which}(V_k == G_k)$ ▷ $\mathcal{I}_{l+1}$ is the set of splits identifying the current face.
    $\{i_j\} \leftarrow \mathcal{I}_{l+1} \setminus \mathcal{I}_l$ ▷ Identify the new split
    $\psi \leftarrow \sum_{m=i_{j-1}}^{i_j} \boldsymbol{\gamma}_m^{\uparrow} / \sum_{m=i_j+1}^{i_{j+1}} \boldsymbol{\gamma}_m^{\uparrow}$
    $\boldsymbol{\nu} \leftarrow (0, \ldots, 0, \underbrace{1, \ldots, 1}_{i_{j-1}, \ldots, i_j}, \underbrace{-\psi, \ldots, -\psi}_{i_j+1, \ldots, i_{j+1}}, 0, \ldots, 0)^{\top}$ ▷ The new normal vector to the face that was just intersected.
    $\boldsymbol{\rho}^{(l+1)} \leftarrow \boldsymbol{\rho}^{(l)} - [(\boldsymbol{\rho}^{(l)})^{\top}\boldsymbol{\nu}]\boldsymbol{\nu}/\|\boldsymbol{\nu}\|_2^2$ ▷ Project $\boldsymbol{\rho}$ on the new face
    $l \leftarrow l + 1$
  end while
end procedure, OUTPUT: a sequence of at most n points $(\mathbf{v}^{(l)})_{l \in \{0,\ldots,n-1\}}$ that defines the Pareto curve as the union of the line segments connecting these points.

Theorem 3. *Algorithm 2 returns the Pareto front for the multi-objective optimization problem* (10) *and has complexity* $O(n^2 \log(n))$.

Indeed the while loop is done at most $n-1$ times and in each loop the most complex operation is the sorting operation with complexity $O(n \log(n))$.

6 BALANCED WORDS

A standard approach for delivering a sequence of rankings while respecting certain desired proportions as much as possible, is to sample from a categorical distribution. For instance Singh and Joachims [30] do a BvN decomposition, then randomly sample from the obtained distribution. In the same vein, Oosterhuis [22] optimizes the parameters of a Plackett-Luce (PL) distribution [24] and directly samples the sequence of rankings from it.

In this section we argue that there is a better way of delivering rankings from a distribution that does not involve stochastic sampling. For example suppose we do coin flips and we have determined that heads and tails should be delivered each 50% of the time. If we do ten coin flips (i.e., stochastic sampling), the probability that we get 5 heads and 5 tails (i.e., that we are fair) is $252/1024 \approx 1/4$. Therefore a better policy is to manually take the coin and lay it alternatively once on the head once on the tail, and so on.

This intuition can be generalized to non-uniform distributions over more than two possible values by using *m-balanced words* [36]. When expressed in the terms of our problem, a generator of *m-balanced words* produces a sequence of rankings such that, in any pair of sub-strings with identical length, the frequency of any ranking differs at most by m. Such a sequence is called a word and a ranking corresponds to a letter in this word. In other words, this generator guarantees that the generated sequence delivers the rankings with proportions as close as possible to the target ones. An important theoretical fact is that the best achievable m is at most the number of unique letters (i.e., rankings) minus 1 [29]. Another advantage of this approach with respect to a BvN decomposition then becomes apparent: The fact that our distribution is over at most n distinct rankings instead of $(n-1)^2+1$ means that we are able to deliver our distribution in a much more balanced way than with a BvN decomposition, i.e., with $m = n-1$ instead of $m = (n-1)^2$. An algorithm capable of efficiently generating m-balanced words, given a certain distribution of letters is given in Algorithm 1 of [29] and in Appendix B. This generator is equivalent to the well-known *Stride Scheduling* algorithm, used to generate fair sequences in resource (CPU) management for concurrent processes [37].

7 EXPERIMENTS

In order to empirically verify our claims, we perform experiments on both synthetic data and on publicly available datasets. Our source code is available on github[3]. We executed the computations on a laptop with an Intel®Core™i7-8650U CPU @ 1.90GHz processor.

We use the TREC2020 Fairness Track evaluation queries and items [2] for which we computed relevance probabilities ourselves. The computed values as well as details about the method we used to compute them will be made available in our github repository as they are not of primary importance for this paper. We also use the MSLR dataset [25] for which we used the ground truth relevances graded from 0 (worst grade) to 4 (best grade), in order to check the influence of having discrete-valued relevance values. We normalize them to the range $[0, 1]$ by dividing them by 4.

For both datasets we restrain ourselves to queries having fewer than 100 items, because our LP baseline would take too much time on these queries and because of some numerical issues in our implementation, discussed in more detail in Appendix A. Furthermore we eliminate uninteresting queries having only one document and queries for which all relevance values are equal. This leaves 795 amongst 867 queries for MSLR and 198 amongst 200 queries for TREC.

[3] https://github.com/tillkletti/expohedron

7.1 Metrics and baselines

For the exposure model we use the exposure vector $\boldsymbol{\gamma}$ of DCG whose k^{th} element is $1/\log_2(k+2)$. The metric measuring the utility is nDCG, which divides the DCG by the "ideal" DCG obtained with PRP rankings only. To aggregate results over several queries, we compute the arithmetic mean of the nDCGs. This indicates for a given ranking policy, what percentage of the maximal utility we achieve on average.

We consider the relevance values to be the merit of the items. By default, we define the target exposure as $\frac{\|\boldsymbol{\gamma}\|_1}{\|\boldsymbol{\rho}\|_1}\boldsymbol{\rho}$, so that the sum of exposures has the right scale. When the target exposure happens not to be inside the expohedron, this definition leads to infeasible target exposures. In that case, we add a constant to the merits of all documents until the target becomes feasible, by choosing the smallest value $b \in \mathbb{R}_+$ such that $\boldsymbol{\mathcal{E}}^* := (1-b)\frac{\|\boldsymbol{\gamma}\|_1}{\|\boldsymbol{\rho}\|_1}\boldsymbol{\rho} + b\frac{\|\boldsymbol{\gamma}\|_1}{n}\mathbf{1} \prec \boldsymbol{\gamma}$. This can be efficiently computed using (12), since it corresponds to finding the intersection of the line segment $\left[\frac{\|\boldsymbol{\gamma}\|_1}{\|\boldsymbol{\rho}\|_1}\boldsymbol{\rho}, \frac{\|\boldsymbol{\gamma}\|_1}{n}\mathbf{1}\right]$ with the border of the expohedron. To aggregate results over several queries, we average a *normalized unfairness*: $\frac{\|\boldsymbol{\mathcal{E}}-\boldsymbol{\mathcal{E}}^*\|_2}{\|\boldsymbol{\gamma}\|_1}$ to bring the exposures to the same scale. We evaluate our own method and 4 baselines, in their ability to solve our MOO problem, both in terms of effectiveness and efficiency:

(1) Our own Expohedron (Expo) method with our own Carathéodory decomposition (GLS) delivered using an m-balanced word (BW) generator as described in [29]. In a variant (expo end), we compute only the fairness endpoint of the Pareto-front.
(2) A Plackett-Luce (PL) distribution [24] with parameters $\boldsymbol{\rho}$ and with a varying temperature parameter τ. The temperature parameter controls the interpolation between PRP rankings and a uniform distribution over all rankings.
(3) A Linear Program (LP) approach as proposed by [30]. There is no trade-off between relevance and fairness with this approach; only an endpoint of the Pareto front is found, corresponding to zero unfairness, called *fairness endpoint*. We use the `cvxopt` solver [1] to solve the LP and we use a BvN decomposition implemented in [35].
(4) A Quadratic Program (QP) that finds bistochastic matrices maximizing, for varying α, the scalarized MOO
$$\max_{\pi \in \text{Conv}(\mathcal{S}_n)} \alpha \boldsymbol{\rho}^\top \pi \boldsymbol{\gamma} - (1-\alpha)\left\|\pi\boldsymbol{\gamma} - \boldsymbol{\mathcal{E}}^*\right\|_2^2. \tag{13}$$
(5) A Controller (Ctrl) as proposed by [34] with parameter $K = 1$ and for varying gain, hoping to find approximately Pareto-optimal solutions. This controller computes at each time-step whether an item is disadvantaged and tries to compensate this in future rankings by increasing its relevance with a small bonus. Then a PRP ranking is delivered using the modified relevances.

7.2 Results

How does the runtime of our method and of the baselines vary when the number of items increases ? In order to fully control the number of items, we use synthetic data for this research question. For each $n \in \{2, \dots, 100\}$, we sample 100 random relevance vectors whose elements are uniformly independently distributed in $[0, 1]$.

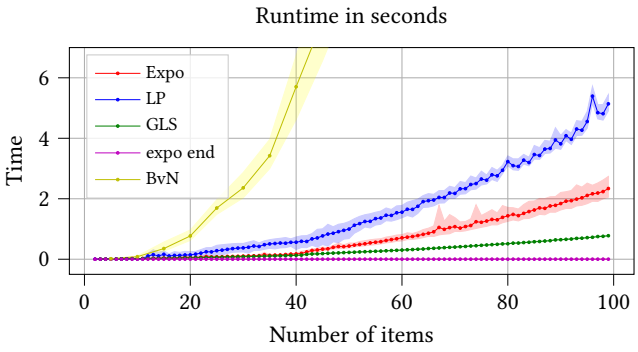


Figure 4: Runtime of different algorithmic modules as a function of the number of items n. For each evaluated n, 100 random relevance vectors are sampled uniformly and independently. The shaded areas correspond to 95% quantiles. The yellow curve corresponding to the BvN leaves the frame, reaching 70 seconds by $n = 100$. For a global comparison, the sum of the expo end and GLS curves should be compared with the sum of the LP and BvN curves.

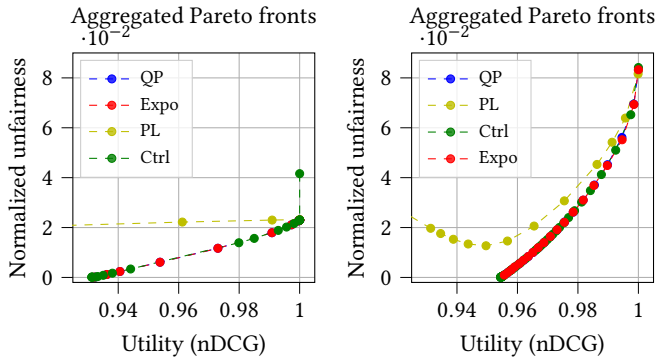
For each of those n, we compute the full Pareto front (expo) in the expohedron. We compute the bistochastic matrices corresponding to the fairness endpoint of the point using LP and compute its BvN decomposition. For a fair comparison with LP, we derive the fairness endpoint for each n using our Expo method (expo end), and its Carathéodory decomposition with Algorithm 1 (GLS).

The average times the different algorithmic components take are reported in Figure 4. It appears that the BvN decomposition is particularly slow, while the computation of the fairness endpoint using our Expo method is almost instantaneous. In particular it appears that the advantage of our method (fairness endpoint computation + GLS decomposition) w.r.t. an LP approach (including the BvN decomposition) increases with increasing number of items.

How far are the baseline methods from complete Pareto optimality? Among our baselines, QP is guaranteed to produce Pareto-optimal points, provided that the associated QP solver has sufficient precision. For both the TREC and the MSLR datasets, we generate points of all Pareto-fronts (one front per query) by varying the trade-off parameter α in (13) between 0 and 1. To plot utility-unfairness points corresponding to the PL approach, we compute for each query the performance of PL after $T = 1\,000$ rankings, with the temperature parameter τ varying in $[1 \cdot 10^{-3}, 1]$ for TREC and in $[1 \cdot 10^{-5}, 10]$ for MSLR. The performances corresponding to the same temperature are aggregated over the different queries with the aggregation method described in 7.1.

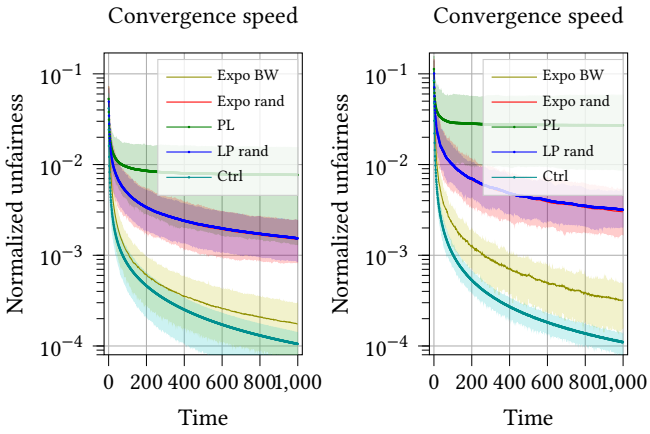
For Ctrl, we generate points in the same utility-unfairness space by varying the gain within $[0, 1\,000]$ for TREC and within $[0, 10\,000]$ for MSLR. The performances corresponding to the same gain are aggregated over the different queries to produce the plotted points.

Our Expo method does not produce a parametrized set of utility-unfairness solutions. Knowing that our Expo method should theoretically produce the same solutions as QP for some value of α, we check that every solution of the QP, expressed as an exposure vector after multiplying the optimal bistochastic matrix by the $\boldsymbol{\gamma}$ vector, corresponds to a point of the Expo-derived Pareto-front and associate the corresponding α to the latter. We are then able to aggregate the performances over the different queries, by considering the solutions associated to the same α.



(a) MSLR (b) TREC

Figure 5: Aggregated Pareto fronts. The red and blue points are overlaid along the Pareto front and therefore hard to distinguish.



(a) MSLR (b) TREC

Figure 6: Average normalized unfairness as a function of the number of delivered rankings. The red and blue curves are almost equal and may be difficult to distinguish. The shaded areas represent 95% quantiles.

The results are reported in Figure 5. It appears without surprise that both QP and Expo perform identically, while PL is far from Pareto-optimality, except in the region where nDCG reaches 1. More surprisingly Ctrl performs well for a heuristic approach, with its points being indistinguishable from the actual Pareto front, except for an outlier obtained with gain = 0 for the MSLR dataset. However, as it does not explicitly address the problem as a MOO problem and as it uses a parameter (the gain) which controls very indirectly the utility-fairness trade-off, there is as of yet no guarantee that it will always result in near-optimal Pareto-fronts in all settings.

How do the objectives evolve over time ? Does using balanced words provide an advantage w.r.t. sampling ? For each method we select the parameter that brings it closest to minimal unfairness. For Expo this corresponds to simply choosing the fairness endpoint. For PL we select $\tau = 1$. For Ctrl we select a gain of 0.6 for TREC and a gain of 10 for MSLR. For LP there is no parameter to set and we do not apply a QP, since LP does the same job more efficiently. We aggregate the results over all queries by taking the average normalized unfairness at each time-step.

The results are reported in Figure 6. It appears that sampling from a distribution found via BvN or via GLS has no notable effect on transient performance. However using balanced words instead of sampling does have a beneficial effect on the speed of convergence to the target exposure. It also appears that Ctrl does converge more quickly than all other methods, at least for the fairness endpoint.

Is Expo faster than existing baselines ? In Table 1 we report the time it took to compute the solutions using our Expo method and the baselines. It appears that Expo is clearly superior to the other exact methods LP and QP in terms of runtime. For $T = 1\,000$ rankings Ctrl is quicker than Expo. However for $T = 10\,000$ rankings, Ctrl takes ten times as long, whereas for the Expo method, the computation of the Pareto-front (or endpoint) and GLS need to be performed but once. Then once a solution is found, it can be delivered very quickly using balanced words. Thus for larger time horizons with an order of magnitude of several thousand rankings, Expo becomes quicker than Ctrl for the total runtime.

Table 1: Average runtimes in seconds for each dataset for $T = 1\,000$ and $T = 10\,000$ rankings.

	TREC $T = 1\,000$	MSLR $T = 1\,000$	TREC $T = 10\,000$	MSLR $T = 10\,000$
Expo endpoint	0.0014	0.0018	0.0014	0.0018
Expo	0.0806	0.0199	0.0806	0.0199
LP	0.0858	2.1369	0.0858	2.1369
QP	0.8424	15.2095	0.8424	15.2095
GLS	0.0737	0.5396	0.0737	0.5396
BvN	2.2623	29.1706	2.2623	29.1706
BW	0.0009	0.0010	0.0094	0.0093
rand	0.0388	0.0469	0.4060	0.4340
Expo+GLS+BW	0.1552	0.5605	0.1637	0.5688
LP+BvN+rand	2.3869	31.3544	2.7541	31.7415
Ctrl	0.0245	0.0268	0.2201	0.2543
PL	0.0145	0.0174	0.1525	0.1991

8 CONCLUSION

Our novel geometrical framework makes it possible to efficiently compute all Pareto-optimal fairness-utility amortizations for a PBM. Amongst the methods that are provably Pareto-optimal, our method is the overall quickest. The controller from [34] empirically performs equally well and is quicker for time horizons lower than several thousands.

In future work we plan to extend our framework to group fairness and to different exposure models that are not PBMs, such as Dynamic Bayesian Network models [6].

ACKNOWLEDGMENTS

This work has been partially supported by MIAI@Grenoble Alpes, (ANR-19-P3IA-0003).

REFERENCES

[1] 2021. cvxopt/cvxopt. https://github.com/cvxopt/cvxopt original-date: 2013-02-22T22:14:31Z.

[2] Asia J. Biega, Fernando Diaz, Michael D. Ekstrand, and Sebastian Kohlmeier. 2020. Overview of the TREC 2020 Fair Ranking Track. In *The Twenty-Eighth Text REtrieval Conference (TREC 2020) Proceedings*.

[3] Asia J. Biega, Krishna P. Gummadi, and Gerhard Weikum. 2018. Equity of Attention: Amortizing Individual Fairness in Rankings. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. 405–414.

[4] C. Carathéodory. 1907. Über den Variabilitätsbereich der Koeffizienten von Potenzreihen, die gegebene Werte nicht annehmen. *Math. Ann.* 64, 1 (March 1907), 95–115.

[5] Olivier Chapelle and Ya Zhang. 2009. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web (WWW '09)*. 1–10.

[6] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click Models for Web Search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 7, 3 (July 2015), 1–115.

[7] Fernando Diaz, Bhaskar Mitra, Michael D. Ekstrand, Asia J. Biega, and Ben Carterette. 2020. Evaluating Stochastic Rankings with Expected Exposure. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM '20)*. 275–284.

[8] Virginie Do, Sam Corbett-Davies, Jamal Atif, and Nicolas Usunier. 2021. Online certification of preference-based fairness for personalized recommender systems. *arXiv:2104.14527 [cs, stat]* (April 2021). arXiv: 2104.14527.

[9] Fanny Dufossé and Bora Uçar. 2016. Notes on Birkhoff–von Neumann decomposition of doubly stochastic matrices. *Linear Algebra Appl.* 497 (May 2016), 108–115.

[10] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through Awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference* (Cambridge, Massachusetts) *(ITCS '12)*. 214–226.

[11] Sahin Cem Geyik, Stuart Ambler, and Krishnaram Kenthapadi. 2019. Fairness-Aware Ranking in Search & Recommendation Systems with Application to LinkedIn Talent Search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. 2221–2231.

[12] Martin Grötschel, László Lovász, and Alexander Schrijver. 1993. *Geometric Algorithms and Combinatorial Optimization* (2 ed.). Springer-Verlag, Berlin Heidelberg.

[13] Ruben Hoeksma, Bodo Manthey, and Marc Uetz. 2016. Efficient implementation of Carathéodory's theorem for the single machine scheduling polytope. *Discrete Applied Mathematics* 215 (Dec. 2016), 136–145.

[14] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.* 20, 4 (Oct. 2002), 422–446.

[15] M. Marcus and R. Ree. 1959. Diagonals of doubly stochastic matrices. *The Quarterly Journal of Mathematics* 10, 1 (Jan. 1959), 296–302.

[16] Albert W. Marshall, Ingram Olkin, and Barry C. Arnold. 2011. *Inequalities: Theory of Majorization and Its Applications*. Springer New York, New York, NY.

[17] Alistair Moffat and Justin Zobel. 2008. Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems* 27, 1 (Dec. 2008), 2:1–2:27.

[18] Marco Morik, Ashudeep Singh, Jessica Hong, and Thorsten Joachims. 2020. Controlling Fairness and Bias in Dynamic Learning-to-Rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 429–438.

[19] Harikrishna Narasimhan, Andrew Cotter, Maya Gupta, and Serena Wang. 2020. Pairwise Fairness for Ranking and Regression. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI '21)*. 5248–5255.

[20] Márton Naszódi and Alexandr Polyanskii. 2019. Perron and Frobenius meet Carathéodory. *arXiv:1901.00540 [math]* (Jan. 2019). arXiv: 1901.00540.

[21] Shmuel Onn. 1993. Geometry, complexity, and combinatorics of permutation polytopes. *Journal of Combinatorial Theory, Series A* 64, 1 (Sept. 1993), 31–49.

[22] Harrie Oosterhuis. 2021. Computationally Efficient Optimization of Plackett-Luce Ranking Models for Relevance and Fairness. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 1023–1032.

[23] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. 2021. Fairness in rankings and recommendations: an overview. *The VLDB Journal* (Oct. 2021).

[24] R. L. Plackett. 1975. The Analysis of Permutations. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 24, 2 (1975), 193–202.

[25] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013). http://arxiv.org/abs/1306.2597

[26] R. Rado. 1952. An Inequality. *Journal of the London Mathematical Society* s1-27, 1 (1952), 1–6.

[27] Stephen Robertson. 1977. The Probability Ranking Principle in IR. *Journal of Documentation* 33 (Dec. 1977), 294–304.

[28] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. 2019. Interventional Fairness: Causal Database Repair for Algorithmic Fairness. In *Proceedings of the 2019 International Conference on Management of Data* (Amsterdam, Netherlands) *(SIGMOD '19)*. 793–810.

[29] Shinya Sano, Naoto Miyoshi, and Ryohei Kataoka. 2004. m-Balanced words: A generalization of balanced words. *Theoretical Computer Science* 314, 1-2 (Feb. 2004), 97–120.

[30] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of Exposure in Rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. 2219–2228.

[31] Ashudeep Singh and Thorsten Joachims. 2019. Policy Learning for Fairness in Ranking. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc.

[32] Ashudeep Singh, David Kempe, and Thorsten Joachims. 2021. Fairness in Ranking under Uncertainty. *arXiv:2107.06720 [cs]* (July 2021). arXiv: 2107.06720.

[33] Yi Su, Magd Bayoumi, and Thorsten Joachims. 2021. Optimizing Rankings for Recommendation in Matching Markets. *arXiv:2106.01941 [cs]* (June 2021). arXiv: 2106.01941.

[34] Thibaut Thonet and Jean-Michel Renders. 2020. Multi-grouping Robust Fair Ranking. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 2077–2080.

[35] Brandon Trabucco. 2021. brandontrabucco/bvn. https://github.com/brandontrabucco/bvn original-date: 2020-09-15T00:11:02Z.

[36] Laurent Vuillon. 2003. Balanced words. *Bulletin of the Belgian Mathematical Society - Simon Stevin* 10, 5 (Dec. 2003), 787–805.

[37] C. A. Waldspurger and E. Weihl. W. 1995. *Stride Scheduling: Deterministic Proportional- Share Resource Management.* Technical Report. Massachusetts Institute of Technology, USA.

[38] Lequn Wang, Yiwei Bai, Wen Sun, and Thorsten Joachims. 2021. Fairness of Exposure in Stochastic Bandits. *arXiv:2103.02735 [cs]* (March 2021). arXiv: 2103.02735.

[39] Lequn Wang and Thorsten Joachims. 2021. User Fairness, Item Fairness, and Diversity for Rankings in Two-Sided Markets. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '21)*. 23–41.

[40] Yao Wu, Jian Cao, Guandong Xu, and Yudong Tan. 2021. TFROM: A Two-sided Fairness-Aware Recommendation Model for Both Customers and Providers. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 1013–1022.

[41] Shota Yasutake, Kohei Hatano, Shuji Kijima, Eiji Takimoto, and Masayuki Takeda. 2011. Online Linear Optimization over Permutations. In *Algorithms and Computation (Lecture Notes in Computer Science)*, Takao Asano, Shin-ichi Nakano, Yoshio Okamoto, and Osamu Watanabe (Eds.). Springer, Berlin, Heidelberg, 534–543.

[42] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. FA*IR: A Fair Top-k Ranking Algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* (Singapore, Singapore) *(CIKM '17)*. 1569–1578.

[43] Meike Zehlike and Carlos Castillo. 2020. Reducing Disparate Exposure in Ranking: A Learning To Rank Approach. In *Proceedings of The Web Conference 2020 (WWW '20)*. 2849–2855.

APPENDICES

A Implementation

Our algorithms 1 and 2 are exact algorithms on paper, but in practice, when implemented in a machine that cannot handle real numbers, some numerical difficulties are encountered. In this section we explain some tricks we used to tackle these difficulties and their limitations.

Consider Algorithm 1. When doing a bisection search with a given precision on a half-line, in order to find its intersection with the border of the expohedron, the point $\mathbf{p}$ that is found only approximately lies on the intersected face. Therefore the next half-line starting at a vertex $\mathbf{v}$ and passing through $\mathbf{p}$ is not exactly contained in the subspace containing the face of $\mathbf{v}$ and $\mathbf{p}$. The further away a point on the half-line is from $\mathbf{v}$, the bigger this error will be. In the worst case the error will be big enough so that the next intersection with the border of the expohedron is not on a face of lower dimension, thereby preventing convergence of the algorithm. Such a problem can be avoided by projecting the approximate point $\mathbf{p}$ on the hyperplane containing the face on which $\mathbf{p}$ lies. This substantially reduces the error of $\mathbf{p}$ and makes it possible to avoid many convergence problems.

Our implementation still encounters convergence problems when the number of documents is high. In that case the expohedron is a very high-dimensional polytope and some faces are smaller than the implementation's precision, which makes face-identification error-prone by the end of Algorithm 2, when it is already close to maximum utility. This issue can be avoided by putting a tolerance in line 7 of Algorithm 2. Typically for our experiments, setting a tolerance of 10^{-6} was sufficient.

B Balanced words

We lay out here Algorithm 3 to efficiently generate m-balanced words [29]. In practice we let let the algorithm warm up for a few hundred iterations, when $\boldsymbol{\phi}$ is initialized to $\mathbf{0}$. Otherwise the algorithm starts by delivering each ranking once, regardless of its density.

Algorithm 3 A generator of m-balanced words

1: **procedure** BW(INPUT: a density $\alpha_1, \dots, \alpha_N \in \mathbb{R}_+$, $\sum_{i=1}^N \alpha_i = 1$ and an alphabet $\mathcal{A}$ with elements $(a_i)_{i\in\{1,\dots,N\}}$)
2: $\quad\boldsymbol{\phi} \leftarrow \mathbf{0} \in \mathbb{R}^N$
3: $\quad t \leftarrow 0$
4: $\quad$**while** True **do** ▷ Generate the next element of the sequence
5: $\quad\quad i^* \leftarrow \arg\min_{i\in\{1,\dots,N\}} \boldsymbol{\phi}_i$
6: $\quad\quad \mathbf{x}_t \leftarrow a_{i^*}$
7: $\quad\quad \boldsymbol{\phi}_{i^*} \leftarrow \boldsymbol{\phi}_{i^*} + \frac{1}{\alpha_{i^*}}$
8: $\quad\quad t \leftarrow t + 1$
9: $\quad$**end while**
10: **end procedure** OUTPUT: A sequence $\mathbf{x} \in \mathcal{A}^{\mathbb{N}}$. $\mathbf{x}$ is an $N-1$-balanced word.