Fairness and Bias in Online Selection

José Correa [1] Andrés Cristi [1] Paul Dütting [2] Ashkan Norouzi-Fard [2]

Abstract

There is growing awareness and concern about fairness in machine learning and algorithm design. This is particularly true in online selection problems where decisions are often biased, for example, when assessing credit risks or hiring staff. We address the issues of fairness and bias in online selection by introducing multi-color versions of the classic secretary and prophet problem. Interestingly, existing algorithms for these problems are either very unfair or very inefficient, so we develop optimal fair algorithms for these new problems and provide tight bounds on their competitiveness. We validate our theoretical findings on real-world data.

1. Introduction

The sharp growth in data availability that characterizes modern society challenges our processing capabilities, not only because of its massiveness, but also because of the increasing strict social norms that society seeks in the algorithms processing it. For instance, machine learning algorithms are now used to make credit and lending decisions, to estimate the success of a kidney transplant, to inform hiring decisions, to recommend schools to pupils, among others. Therefore there is a funded concern over the use of algorithms that may violate social norms. Two basic such norms, that are receiving significant attention are fairness and privacy, and while a formalization of the latter is relatively well established through the notion of differential privacy (Dwork et al., 2006), the former is much more unexplored from an algorithmic perspective (Kearns & Roth, 2019).

In this paper we are particularly interested in the study of fairness in machine learning algorithms in the context of sequential decision making under stochastic input. Not only the area has seen many recent theoretical developments, but also it naturally encompasses many real-world decision making processes where biased evaluations should be avoided, such as those mentioned earlier.

Specifically we consider the basic single item selection models given by the secretary and prophet problems in which items are classified into different groups. In our secretary model, two candidates from different groups are incomparable, while in our prophet problem the decision maker is constrained to pick from each group with a prespecified probability. A precursor of the study of fairness in the secretary problem is the work of Buchbinder et al. (2014) who studied, among other things, an incentive compatible version of the secretary problem in which the selection probability does not depend on the arrival position of a candidate. More recently, Gupta & Salem (2020) have studied machine learning algorithms for biased versions of the secretary problem, whereas Cayci et al. (2020) studies similar issues from the perspective of online learning. The fairness term has been used for various concepts in machine learning community. We adopt here the common notion used in various previous works Halabi et al. (2020); Celis et al. (2018a;b;c); Chierichetti et al. (2019; 2017), where we ask that the solution obtained is balanced with respect to some sensitive attribute (e.g., race, gender).

1.1. Our Results

We consider two fundamental problems in fair online selection, both concerned with selecting a single candidate. Candidates are partitioned into different groups or *colors*. The candidates arrive sequentially and upon arrival of a candidate we have to irrevocably decide whether we want to select the candidate or not. In the first problem we consider, which we call the *multi-color secretary problem*, candidates arrive in uniform random order and we can rank candidates within a group, but we cannot compare candidates across groups. There is also a prior probability that the best candidate from a group is the best candidate overall. The problem models situations in which different qualities of the candidates make them largely incomparable (this could arise in some form due to gender, race, social origin, type of education, etc.). The goal is to maximize the probability with which we stop at the best overall candidate and compare it with that for the *offline* optimum. Note that here the offline optimum simply picks the best candidate from the group

*Equal contribution [1]Department of Industrial Engineering, Universidad de Chile, Santiago, Chile. [2]Google Research, Zürich, Switzerland. Correspondence to: Andrés Cristi <andres.cristi@ing.uchile.cl>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

of largest prior probability. Thus, it is extremely unfair. One may think that the best possible online algorithm is to mimick the offline optimum; namely to select the group of largest prior probability and then run the classic secretary algorithm on that group. We prove that this is not the case and indeed our main result is to obtain the best possible online algorithm for the problem and to establish that it satisfies very desirable fairness properties. Hence, for this variant on online selection, fairness follows as a consequence of being online optimal.

In our second problem, which we call the *multi-color prophet problem*, candidates have values drawn independently from given distributions and arrive in an arbitrary order. The goal is to maximize the expectation of the value of the selected candidate, while selecting from each color with probability proportional to a prescribed vector. We compete with *fair opt*, the optimal offline algorithm satisfying the same fairness constraint. So the underlying paradigm here is that although we can compare we understand that the scores may be biased and want to correct the selection process using the prior probabilities. Our main results are the design of fair competitive algorithms. In the most general version we prove that an approximation factor of $1/2$ is best possible while improved factors can be obtained by making natural assumptions on the prior probabilities and the arrival order.

2. The Multi-Color Secretary Problem

In the *multi-color secretary problem* n candidates arrive in uniform random order. Candidates are partitioned into k groups $C = \{C_1, \cdots, C_k\}$. We write $\mathbf{n} = (n_1, \ldots, n_k)$ for the vector of group sizes, i.e., $|C_j| = n_j$, for all $1 \leq j \leq k$. We identify each of the groups with a distinct color and denote by $c(i)$ the color of candidate i. We can compare candidates of the same color, but we cannot compare candidates across groups. We assume comparisons are strict, and use $i \succ i'$ to denote that candidate i is better than candidate i'. We write $\max C_j$ for the best candidate of color j, and $\max C$ for the best candidate overall. A natural assumption is that the best candidate from a group is the best candidate overall with equal probability $1/k$, but we can also consider the case where these probabilities are different. We denote the probabilities with which the best candidate of group j is the best candidate overall by p_j, and write $\mathbf{p} = (p_1, \ldots, p_k)$ for the vector of these probabilities. Since candidates are incomparable across groups this can be modeled by tossing a coin *after the fact* to decide whether the best candidate of group j is the best candidate overall. The goal is to design an online algorithm that maximizes the probability of selecting the best candidate overall.

Even though they do not take a fairness perspective, our model is related to the ones considered by Kumar et al. (2011) and Feldman & Tennenholtz (2012). Kumar et al. (2011) study the problem of selecting a maximal secretary from a partially ordered set of candidates. Feldman & Tennenholtz (2012) study the problem of selecting candidates in parallel: each candidate is randomly assigned to a queue and candidates can only be compared with other candidates in the same queue. The objective is to select a maximal candidate. The two models treat all maximal candidates equally so their results apply only to the case where all p_j's are equal.

2.1. Key Definitions

Competitive ratio. We evaluate online algorithms by means of their competitive ratio. Consider some online algorithm ALG. The algorithm selects the best candidate overall, if it selects the best candidate of a given color and this color has the best candidate overall. For an instance of the multi-color secretary problem with group sizes $\mathbf{n}$ and probabilities $\mathbf{p}$, we denote by $\text{ALG}(\mathbf{n}, \mathbf{p}) \in \{1, ..., n\} \cup \{\phi\}$ the random index at which the algorithm stops, where ϕ denotes the case when the algorithm does not stop. The success probability of ALG is $\mathbf{E}[\mathbf{1}_{\text{ALG}(\mathbf{n},\mathbf{p})=\max C}] = \mathbf{E}[p_{c(\text{ALG}(\mathbf{n},\mathbf{p}))} \cdot \mathbf{1}_{\text{ALG}(\mathbf{n},\mathbf{p})=\max C_{c(\text{ALG}(\mathbf{n},\mathbf{p}))}}]$. We compare this to the optimal offline algorithm OPT, i.e., the best algorithm that can select a candidate after all candidates have arrived. An optimal strategy is to choose a color j with maximum p_j, and then, choose the best candidate of that color. We denote by $\text{OPT}(\mathbf{n}, \mathbf{p}) \in \{1, ..., n\}$ the random index that OPT selects. The success probability of OPT is $\mathbf{E}[\mathbf{1}_{\text{OPT}(\mathbf{n},\mathbf{p})=\max C}] = \max\{p_1, \ldots, p_k\}$.

Definition 1 (competitive ratio)**.** Fix k and $\mathbf{p} = (p_1, \ldots, p_k)$. An online algorithm ALG is $\beta(k, \mathbf{p})$-*competitive* if for all input lengths n and partition sizes $\mathbf{n} = (n_1, \ldots, n_k)$,

$$\frac{\mathbf{E}[\mathbf{1}_{\text{OPT}(\mathbf{n},\mathbf{p})=\max C}]}{\mathbf{E}[\mathbf{1}_{\text{ALG}(\mathbf{n},\mathbf{p})=\max C}]} \leq \beta(k, \mathbf{p}).$$

Note that $\beta(k, \mathbf{p}) \geq 1$, and the smaller $\beta(k, \mathbf{p})$ the better the approximation guarantee.

Unbiased selection. We also examine the extent to which online or offline algorithms are biased, where ideally selection should be unbiased. One way to measure this is by quantifying how much the probability of selecting from any given color class j can differ from the corresponding probability p_j.

Definition 2 (fairness)**.** Fix k and $\mathbf{p} = (p_1, \ldots, p_k)$. An offline or online algorithm ALG is $\alpha(\mathbf{n}, \mathbf{p})$-*fair*, where

$\alpha(\mathbf{n},\mathbf{p}) \geq 1$, if for all colors $j \in [k]$,

$$\frac{p_j}{\alpha(\mathbf{n},\mathbf{p})} \leq \mathbf{P}(c(\mathrm{ALG}(\mathbf{n},\mathbf{p})) = j \mid \mathrm{ALG}(\mathbf{n},\mathbf{p}) \neq \phi) \leq \alpha(\mathbf{n},\mathbf{p}) \cdot p_j .$$

Uniform arrival times. We model uniform random arrival *order* through uniform random arrival *times*. For this, we sample n independent realizations of the Uniform$[0,1]$ distribution, and denote them by $\tau_1 < \tau_2 < \ldots < \tau_n$ indexed in increasing order.

2.2. Optimal Online Algorithm

We derive the optimal online algorithm (without fairness considerations), and observe that—in sharp contrast to the optimal offline algorithm—it is robustly fair and provides an "equal treatment of equals" guarantee.

2.2.1. The Algorithm

We show that the optimal online algorithm is from the class of algorithms given by Algorithm 1. Algorithms from this class receive as input a vector of thresholds $\mathbf{t} = (t_1, \ldots, t_k)$, one for each color $j \in [k]$. When a candidate i arrives the algorithm first checks if the candidate arrived after the time threshold for its color $t_{c(i)}$, and if it did, then it accepts the candidate if it is the best candidate of that color so far.

Algorithm 1 GroupThresholds($\mathbf{t}$)

```
Input: t ∈ [0,1]^k, a threshold in time for each group
Output: i ∈ [n], index of chosen candidate
/* assuming arrival times $\tau_1 < \ldots < \tau_n$ */
for $i \leftarrow 1$ to $n$ do
    if $\tau_i > t_{c(i)}$ then
        if $i \succ \max\{i' \mid \tau_{i'} \leq \tau_i, c(i') = c(i)\}$ then
            return $i$
        end
    end
end
```

Notice that the time based arrival model considered in this section is equivalent to the random order arrival model and is used for the sake of simplicity of presentation and proofs. If we are given an algorithm in the time-based model (such as Algorithm 1), then we can translate it into the random arrival model by having the algorithm draw n arrival times from Uniform$[0,1]$, and assign the i-th smallest arrival time to the i-th candidate in the input stream. If, on the other hand we are given an algorithm in the uniform arrival model, then we can translate it into the time-based model by just ignoring the time component and just using that the candidate that arrived at τ_i was the i-th candidate to arrive.

Therefore any algorithm in one model can be easily used in the other model with identical properties.

2.2.2. Competitive Ratio

Surprisingly, we can show that for any probabilities $\mathbf{p} = (p_1, \ldots, p_k)$ there exist optimal thresholds $\mathbf{t}^* = (t_1^*, \ldots, t_k^*)$ that achieve the best competitive ratio. Later on, we show how these thresholds can be computed explicitly (see Lemma 4). Using these thresholds in Algorithm 1 results in the promised optimal online algorithm. Let us start by presenting the success probability of our algorithm for general probabilities and then for the special case that $\mathbf{p} = (1/k, \ldots, 1/k)$. Afterwards we provide an overview of the proof of these results.

Theorem 1 (competitive ratio, general probabilities)**.** *Fix k and $\mathbf{p} = (p_1, \ldots, p_k)$. Assume wlog that $p_j \geq p_{j+1}$ for all $j < k$. Then there exist thresholds $\mathbf{t}^* = (t_1^*, \ldots, t_k^*)$ such that $t_j^* \leq t_{j+1}^*$ for all $j < k$ that depend only on the number of colors k and the probabilities $\mathbf{p}$ but not on the number of candidates n or the partition sizes $\mathbf{n} = (n_1, \ldots, n_k)$ such that Algorithm 1 with thresholds $\mathbf{t}^*$ succeeds with probability at least*

$$\sum_{j=1}^{k} \int_{t_j^*}^{t_{j+1}^*} \left(\sum_{j'=1}^{j} p_{j'} \right) \frac{T_j^*}{\tau^j} \, d\tau ,$$

where $T_j^* = \prod_{j'=1}^{j} t_{j'}$. For all k and $\mathbf{p} = (p_1, \ldots, p_k)$, no online algorithm can achieve a better competitive ratio in the worst-case over all number of candidates n and partition sizes $\mathbf{n} = (n_1, \ldots, n_k)$.

For the special case where $\mathbf{p} = (1/k, \ldots, 1/k)$ we obtain the following corollary. It shows that in this case we can set a single threshold, and it also provides a simpler-to-parse formula for the competitive ratio.

Corollary 1 (competitive ratio, equal probabilities)**.** *Fix k and $\mathbf{p} = (1/k, \ldots, 1/k)$. Then there exists a single threshold t^* such that Algorithm 1 with thresholds $\mathbf{t}^* = (t^*, \ldots, t^*)$ achieves a competitive ratio of*

$$k^{\frac{1}{k-1}} .$$

This is 2 for $k = 2$, $\sqrt{3}$ for $k = 3$, and $1 + O(\frac{\log k}{k})$ as $k \to \infty$. For all k and $\mathbf{p} = (1/k, \ldots, 1/k)$, no online algorithm can achieve a better competitive ratio in the worst-case over all number of candidates n and partition sizes $\mathbf{n} = (n_1, \ldots, n_k)$.

We note that a bound of $k^{\frac{1}{k-1}} + O(k/n)$ for the special case where $p_j = 1/k$ for all k also follows from (Kumar et al., 2011).

The main difficulty in proving Theorem 1 and Corollary 1 is that in the point-wise optimal online algorithm, which can be obtained by backward induction, thresholds depend on the number of candidates of each color that have already arrived. This dependency leads to a blow-up in algorithm complexity, and complicates the analysis of the success probability. Our high-level approach is to argue that in the worst-case all n_j's are large, and that in this case the point-wise optimal online algorithm is well approximated by the optimal algorithm from the class of algorithms desbribed in Algorithm 1, which simply sets time-dependent thresholds. So we can optimize over these.

A first ingredient in our proof is Lemma 1, which shows that for the class of algorithms in Algorithm 1, for any vector of thresholds $\mathbf{t}$ the worst-case arises when all n_j's are large .

Lemma 1. *Fix the probabilities $\mathbf{p} = (p_1, \ldots, p_k)$ and a vector of thresholds $\mathbf{t} \in [0,1]^k$. For all $j = 1, \ldots k$, the success probability of* $\textsc{GroupThresholds}(\mathbf{t})$ *is decreasing in n_j.*

Our next pair of lemmas, Lemma 2 and Lemma 3, allow us to bound the success probability of the point-wise optimal online algorithm by the limit success probability of the best algorithm which sets time-dependent thresholds (Algorithm 1).

Lemma 2. *Denote by $\mathrm{GT}(\mathbf{p}, \mathbf{t})$ the limit of the success probability of* $\textsc{GroupThresholds}(\mathbf{t})$ *for a given vector of probabilities $\mathbf{p}$. If $\min_j t_j \geq c > 0$, then the success probability of* $\textsc{GroupThresholds}(\mathbf{t})$ *is at most $\mathrm{GT}(\mathbf{p}, \mathbf{t}) + k \cdot (1-c)^z$, where $z = \min_j n_j$.*

Lemma 3. *For any vector of probabilities $\mathbf{p}$ and sizes $\mathbf{n}$, denote by $\textsc{On}(\mathbf{n}, \mathbf{p})$ the optimal success guarantee of an online algorithm. Then for every $\mathbf{p}, \mathbf{n}$ there exists a vector $\mathbf{t}$ such that $\textsc{On}(\mathbf{n}, \mathbf{p}) \leq \mathrm{GT}(\mathbf{p}, \mathbf{t}) + o(1)$, where $\min_j t_j \geq 1/(2e)$.*

The final ingredient is the following pair of lemmas, Lemma 4 and Lemma 5, which solve for the optimal time-dependent thresholds and give a formula for evaluating the limit success probability in terms of these thresholds.

Lemma 4. *Consider a vector $\mathbf{p}$ such that $p_j \geq p_{j+1}$ for all $j < k$. The optimal thresholds $\mathbf{t}^*$ are given by*

$$t_k^* = (1 - (k-1)p_k)^{\frac{1}{k-1}} \ ,$$

$$t_j^* = t_{j+1}^* \cdot \left(\frac{\sum_{r=1}^{j} \frac{p_r}{j-1} - p_j}{\sum_{r=1}^{j} \frac{p_r}{j-1} - p_{j+1}} \right)^{\frac{1}{j-1}} \text{, for } 2 \leq j \leq k-1$$

$$t_1^* = t_2^* \cdot e^{\frac{p_2}{p_1} - 1} \ .$$

Lemma 5. *Consider vectors of probabilities $\mathbf{p}$ and thresholds $\mathbf{t}$, and assume $t_i \leq t_{i+1}$ for all $i < k$. The limit success probability of* $\textsc{GroupThresholds}(\mathbf{t})$ *is given by*

$$\mathrm{GT}(\mathbf{p}, \mathbf{t}) = \sum_{j=1}^{k} \int_{t_j}^{t_{j+1}} \left(\sum_{j'=1}^{j} p_{j'} \right) \frac{T_j}{\tau^j} \, d\tau \ ,$$

where $T_j = \prod_{j'=1}^{j} t_{j'}$.

Putting together these lemmas yields Theorem 1. Their proofs are deferred to the full version of the paper.

2.2.3. Fairness

The optimal offline algorithm is 1-fair for $\mathbf{p} = (1/k, \ldots, 1/k)$, but as soon as probabilities are unbalanced it will choose only from the colors which have maximum p_j. In the worst case, $|p_j - p_{j'}| < \epsilon$ for all j, j', but the optimal offline algorithm is forced to choose from the unique color j which has maximum p_j. We show that in the case where $p_j = 1/k$ for all j, the optimal online algorithm is not exactly 1-fair, but approaches 1-fairness exponentially fast in the minimum group size $\min_j n_j$.

Theorem 2 (fairness result, equal probabilities)**.** *For any k and $\mathbf{p} = (1/k, \ldots, 1/k)$, Algorithm 1 with the optimal single threshold t^* is $1 + O(k^2 (1 - \frac{1}{e})^{\min_j n_j})$-fair.*

Moreover, we show that the optimal online algorithm is robust and degrades gracefully as we move away from perfectly balanced probabilities.

Theorem 3 (fairness result, general probabilities)**.** *Fix k and $\mathbf{p} = (p_1, \ldots, p_k)$. Algorithm 1 with the optimal choice of thresholds $\mathbf{t}^* = (t_1^*, \ldots, t_k^*)$ ensures that if $p_j = p_{j'}$ then $t_j^* = t_{j'}^*$. Moreover, $\mathbf{t}^*$ is a continuous function of $\mathbf{p}$. So if p_j and $p_{j'}$ are close so are t_j^* and $t_{j'}^*$ and so is the probability of selection. More precisely, if $p_j > p_{j'} > (1-\epsilon)p_j$, then $t_{j'}^* > t_j^* > (1-\epsilon)t_{j'}^*$, and furthermore,*

$$\begin{aligned} 0 < \ & \mathbf{P}(\textsc{GroupThresholds}(\mathbf{t}^*) \text{ selects color } j) \\ & - \mathbf{P}(\textsc{GroupThresholds}(\mathbf{t}^*) \text{ selects color } j') < \epsilon. \end{aligned}$$

To exemplify the conclusion of the last theorem consider that we have two colors, say men and women, and that the prior is such that the top candidate is a woman with probability 60% and a man with probability 40%. This translates into having $\epsilon = 1/3$ in the statement of the theorem, which implies that the algorithm will pick a woman at most 33% more often than a man. See Section 4.1 for more examples and empirical validations of these results.

Both Theorem 2 and Theorem 3 follow from analyzing the optimal thresholds in the respective cases, as established in Lemma 4. The complete proofs are deferred to the full version of the paper.

3. The Multi-Color Prophet Problem

We next consider the following *multi-color prophet problem*. In this model n candidates arrive in uniform random order. Candidates are partitioned into k groups $C = \{C_1, \cdots, C_k\}$. We write $\mathbf{n} = (n_1, \ldots, n_k)$ for the vector of group sizes, i.e., $|C_j| = n_j$, for all $1 \leq j \leq k$. We identify each of the groups with a distinct color and let $c(i), v_i$ denote the color and value of candidate i, respectively. The value v_i that is revealed upon arrival of i, and is drawn independently from a given distribution F_i. We use $\mathbf{F} = (F_1, \ldots, F_n)$ to refer to the vector of distributions. We are also given a probability vector $\mathbf{p} = (p_1, \ldots, p_k)$. The goal is to select a candidate in an online manner in order to maximize the expectation of the value of the selected candidate, while selecting from each color with probability proportional to $\mathbf{p}$. We distinguish between the *basic setting* in which p_j is the proportion of candidates that belong to group j, i.e., $p_j = n_j/n$, and the *general setting* in which $\mathbf{p}$ is arbitrary. We compare ourselves with the *fair optimum*, the optimal offline algorithm that respects the p_j's.

3.1. Key Definitions

Fair optimum. We define $\textsc{FairOpt}(n, C, \mathbf{F}, \mathbf{p})$ as the optimal offline algorithm that selects a candidate of group j with probability p_j for all j, and we write $\mathbf{E}[\textsc{FairOpt}(n, C, \mathbf{F}, \mathbf{p})]$ for the expected value it achieves. More precisely, among the class of randomized rules to select a candidate that choose a candidate from each color j with probability p_j, and can observe the realizations of all values, $\textsc{FairOpt}(n, C, \mathbf{F}, \mathbf{p})$ is the one that maximizes the expectation of the value of the selected candidate.

Intuitively, one can think of $\textsc{FairOpt}$ as the limit of the following experiment. We draw m times, with $m >> 1$, an independent sample of the vector $(v_1, \ldots, v_n)$, so we obtain $\{(v_{i,s})_{i=1}^n\}_{s=1}^m$. In each of the vectors we select a candidate $i^*(s)$ so that $\frac{1}{m}\sum_{s=1}^m v_{i^*(s),s}$ is maximized and $i^*(s)$ belongs to color j in $m \cdot p_j$ of the vectors.

Ex-ante relaxation. We denote by q_i the probability with which $\textsc{FairOpt}(n, C, \mathbf{F}, \mathbf{p})$ selects candidate i. Using these probabilities we can obtain the following upper bound on the performance of $\textsc{FairOpt}$, which is known in the prophets literature as the *ex-ante relaxation* (e.g. Feldman et al., 2016),

$$\textsc{ExAnte}(n, C, \mathbf{F}, \mathbf{p}) = \sum_{i=1}^n q_i \cdot \mathbf{E}(v_i \mid v_i \geq F_i^{-1}(1 - q_i)).$$

Fair selection. We say that an online algorithm ALG is fair if it selects a candidate of each color j with probability proportional to p_j.

Definition 3 (fair online algorithm)**.** We say that an online algorithm ALG is fair if

$$\mathbf{P}(c(\text{ALG}) = j \mid \text{ALG stops}) = p_j \quad \forall 1 \leq j \leq k.$$

Note that this is analogous to being 1-fair in Definition 2.

Approximation ratio. Our goal is to find the fair online algorithm $\textsc{FairAlg}$, with best possible approximation ratio with respect to $\textsc{FairOpt}$. To formally define this, let $\mathbf{E}[\textsc{FairAlg}(n, C, \mathbf{F}, \mathbf{p})]$ denote the expected value achieved by $\textsc{FairAlg}$.

Definition 4 (approximation ratio)**.** We say that online algorithm $\textsc{FairAlg}$ provides an α-approximation if

$$\sup_{n, C, \mathbf{F}, \mathbf{p}} \frac{\mathbf{E}[\textsc{FairOpt}(n, C, \mathbf{F}, \mathbf{p})]}{\mathbf{E}[\textsc{FairAlg}(n, C, \mathbf{F}, \mathbf{p})]} \leq \alpha.$$

Note that the smaller $\alpha \geq 1$ the better. Specifically, if $\alpha = 1$, then the expected value achieved by the fair online algorithm matches that of the fair offline algorithm.

3.2. Optimal Online Algorithms

We develop optimal fair online algorithms with surprisingly small competitive ratio under different assumptions on the setting. In the first setting (Section 3.2.1) we consider an arbitrary fixed order of the candidates, non-identical distributions, and general probabilities $\mathbf{p}$. In the second, we assume that variables are i.i.d., and make the natural additional assumption that the p_j's are proportional to the group sizes (Section 3.2.2). In the third setting we relax the i.i.d. assumption to hold only within groups, and assume that candidates arrive in uniform random order. The analysis of this setting is deferred to the full version of the paper.

Our high-level approach is the following: We design online algorithms that accept each candidate i with probability $\alpha \cdot q_i$, where $\mathbf{q} = (q_1, \ldots, q_n)$ are the marginal probabilities with which the optimal fair offline algorithm $\textsc{FairOpt}$ accepts candidate $i = 1, \ldots, n$. Note that for a fixed choice of α this uniquely determines thresholds $\mathbf{t} = (t_1, \ldots, t_n)$ that we have to set for candidate $i = 1, \ldots, n$. We are still free to choose the parameter α, and we choose it to optimize the worst-case approximation ratio.

Intuitively, choosing a smaller α makes us accept less frequently, but conditional on stopping we choose higher values. The right trade-off between these two forces and hence the right choice of α turns out to be different in each of the three settings. We find that in each of the three settings the optimal approximation ratio is equal to $1/\alpha^*$ where α^* is the optimal choice of α.

3.2.1. General Distributions

We start by considering the setting in which candidates arrive in any fixed order, candidate values are drawn from not-necessarily identical distributions, and the probabilities $\mathbf{p}$ can be arbitrary.

Our algorithm for this case (Algorithm 2) receives as input the probabilities $q_1, \ldots, q_n$ with which $\textsc{FairOpt}$ accepts candidate $1, \ldots, n$. It then sets thresholds so that it accepts each of the candidates with probability $q_i/2$.

Note that for $i = 1$ we can achieve this by setting the threshold to $t_1 = F_1^{-1}(1 - q_1/2)$. For $i = 2$ we have to set a slightly lower threshold than $F_2^{-1}(1 - q_2/2)$, because with some probability namely $q_1/2$ we stop at $i = 1$. Indeed, if we set the threshold to $t_2 = F_2^{-1}(1 - \frac{q_2/2}{1-q_1/2})$ we reach candidate $i = 2$ with probability $1 - q_1/2$ and conditional on reaching it we accept it with probability $\frac{q_2/2}{1-q_1/2}$, so we accept it with probability exactly $q_i/2$ as desired. Continuing like this yields the thresholds used in the algorithm.

Algorithm 2 FAIR GENERAL PROPHET

```
Input: Distributions $F_1, \cdots, F_n$, and $q_1, \cdots, q_n$
Output: $i \in [n]$, index of chosen candidate

$s \leftarrow 0$
for $i \leftarrow 1$ to $n$ do
    if $v_i \geq F_i^{-1}(1 - \frac{q_i/2}{1-s/2})$ then
        return $i$
    end
    $s \leftarrow s + q_i$
end
```

We show that this algorithm is fair, and that it achieves an optimal approximation guarantee. Fairness follows quite directly from the fairness of FAIROPT, because our algorithm accepts with the same marginal probabilities just scaled down by $1/2$. For the approximation guarantee we compare the expected value collected by the online algorithm to the expected value achieved by the ex-ante relaxation, which is constrained to use the marginal probabilities $\mathbf{q}$ of FAIROPT. Since the latter is only higher than the expected value achieved by FAIROPT, it also implies an approximation guarantee with respect to FAIROPT.

Theorem 4. *For general settings and general distributions, Algorithm 2 is fair and achieves a 2-approximation to* FAIROPT. *No fair online algorithm can achieve a better approximation ratio.*

We remark that the bound of 2 in this theorem is incomparable to the well known factor 2 in the regular prophet inequality (Samuel-Cahn, 1984), and indeed we prove it via a substantially different technique.

3.2.2. I.I.D. DISTRIBUTIONS

We next consider the i.i.d. setting, where all values v_i are independent samples from a common distribution F. In this case $p_j = n_j/n$ for all groups j is a natural assumption, because this is the probability with which the maximum overall is from group j. Also note that in this case the optimal offline algorithm is fair, and chooses each element with probability $1/n$. That is, $q_i = 1/n$ for all i.

Our Algorithm 3 tries to mimic the optimal fair offline algorithm, but aims at slightly lower marginal acceptance probabilities of $2/(3n)$. The derivation of the thresholds $\mathbf{t}$ that achieve this follows the same logic as in our algorithm for general distributions.

Algorithm 3 FAIR IID PROPHET

```
Input: Distributions $F$
Output: $i \in [n]$, index of chosen candidate

for $i \leftarrow 1$ to $n$ do
    if $v_i \geq F^{-1}(1 - \frac{2/3n}{1-2(i-1)/3n})$ then
        return $i$
    end
end
```

We prove that this algorithm is fair, and achieves an optimal approximation ratio of $3/2$. To show fairness we again exploit that the algorithm accepts each candidate i with a scaled-down version of the marginal probability $q_i = 1/n$ with which FAIROPT accepts a candidate. We establish the approximation factor via a stochastic dominance argument.

Theorem 5. *For basic settings and i.i.d. distributions Algorithm 3 is fair and achieves a $3/2$-approximation to* FAIROPT. *No fair online algorithm can achieve a better approximation ratio.*

The proofs of Theorem 4 and Theorem 5 and the discussion of the third setting for which we prove a tight approximation ratio of $1/(2 - \sqrt{2}) \approx 1.707$ are deferred to the full version of the paper. In Section 4.2 we experimentally validate these results and compare them with other algorithms in the literature.

4. Empirical Evaluation

In this section we empirically validate our results on synthetical and real-world experiments[1]. We present experiments for the multi-color secretary problem in Section 4.1 and the multi-color prophet problem in Section 4.2.

4.1. Secretary Experiments

We compare our algorithm (Algorithm 1) with the following two baselines, which are based on the optimal solution to the classic secretary problem (e.g. Lindley, 1961; Dynkin, 1963; Ferguson, 1989):

1. Secretary algorithm (SA): This algorithm first computes the maximum value in the first $1/e$-fraction of the stream, and then picks any element with higher value afterwards. This algorithm does not consider the colors of elements.

[1]An implementation of these experiments is available at https://github.com/google-research/google-research/tree/master/fairness_and_bias_in_online_selection.

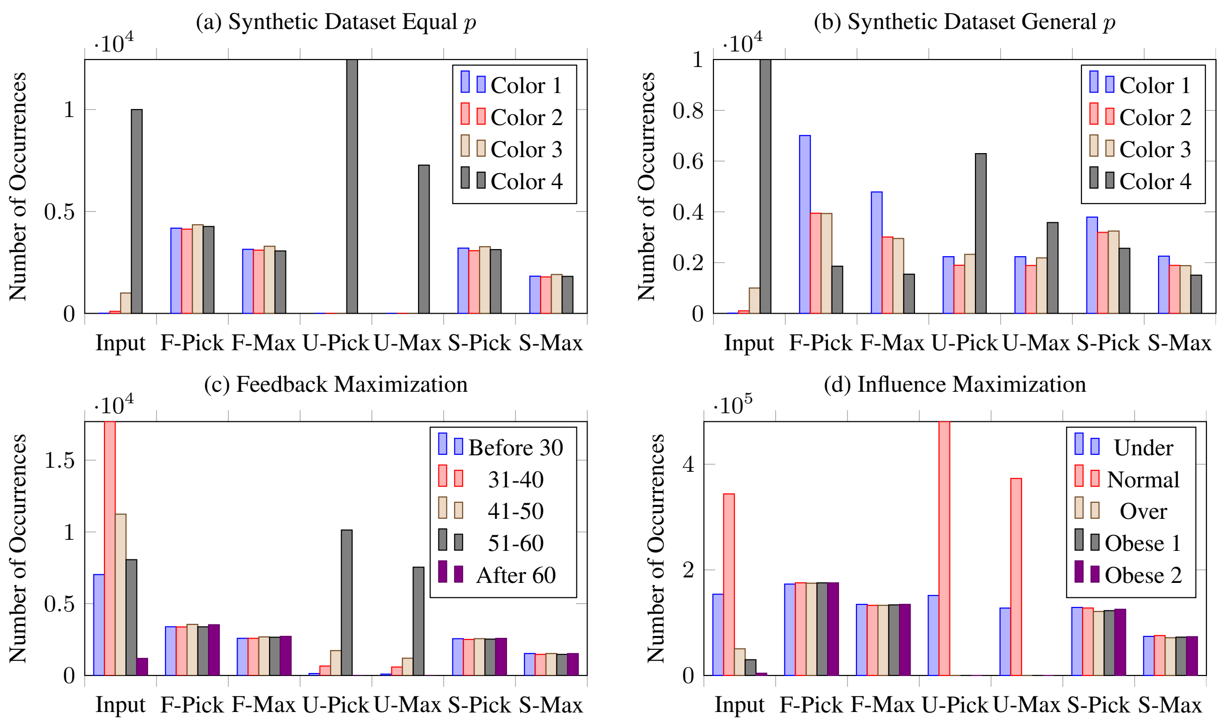


Figure 1. In this plot, we compare our fair secretary algorithm with the secretary algorithm (SA) and the single-color secretary algorithm (SCSA) on (a) synthetic dataset, equal $\mathbf{p}$ values, (b) synthetic dataset, general $\mathbf{p}$ values, (c) feedback maximization dataset, and (d) influence maximization dataset. Here Input is the number of elements from each color in the input, F-Pick and F-Max are the number of elements picked by our fair secretary algorithm and the number of them that are the maximum among the elements of that color. Similarly, U-Pick (S-Pick) and U-Max (S-Max) are the number of elements picked by SA and SCSA and the number of them that are the maximum among the elements of that color.

2. Single-color secretary algorithm (SCSA): This algorithm first picks a color proportional to the $\mathbf{p}$ values, and then runs the secretary algorithm on the elements of that color. This algorithm does not consider the elements whose color is different from the chosen one.

For all the experiments in this section, we run all the algorithms $20,000$ times. We report the number of times that i) the algorithm selects an element from each of the colors, ii) the number of times the selected element has the highest value in its color.

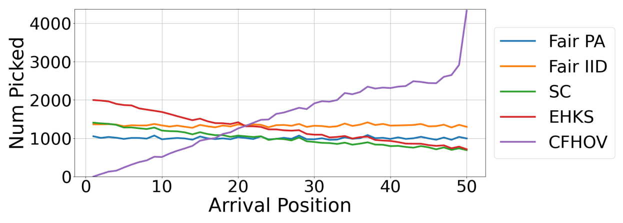
Synthetic dataset, equal $\mathbf{p}$ values. In this experiment, we create a synthetic dataset as follows. There are four colors with $10, 100, 1000$, and 10000 occurrences. The value of each element is chosen independently and uniformly at random from $[0, 1]$, so the p values are the same for all the colors, i.e., $\mathbf{p} = (1/4, 1/4, 1/4, 1/4)$. In Figure 1 (a), we present the result for this dataset. We observe that our algorithm and SCSA pick almost equal number of times from each color,[2] while SA picks almost only from the forth color.

[2]The slight difference is due to the random nature of the algorithm.

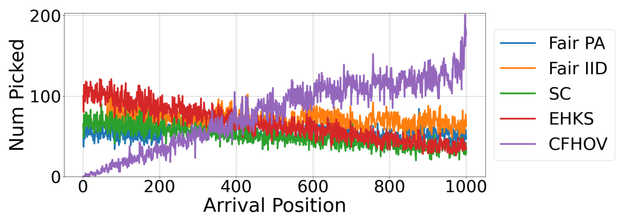
Therefore both our algorithm and SCSA are fair while SA does not satisfy the fairness expectations. We also observe that the number of elements picked by our algorithm is 1.305 times higher than in SCSA (+30.5%), and it picks the maximum element of the color and hence the best element overall 1.721 times more often than SCSA (+73.1%). Therefore the quality of the solution of our algorithm is significantly higher than that of SCSA.

Synthetic dataset, general $\mathbf{p}$ values. In this experiment, we create a synthetic dataset with four colors of sizes $10, 100, 1000, 10000$ and $\mathbf{p} = (0.3, 0.25, 0.25, 0.2)$.. The results are presented in Figure 1 (b). We observe that both the distributions of the picked element for our algorithm and SCSA is close to the p distribution while for SA it is clearly different. Moreover, our algorithm performs significantly better than SCSA since it picks 1.309 times more elements (+30.9%) and 1.630 times more maximum element of the picked color (+63.0%).

Feedback maximization. We consider a dataset containing one record for each phone call by a Portuguese banking institution (Moro et al., 2014). The goal of this experiment is to select a client and contact them and ask for their feed-



(a) Uniform Distribution



(b) Binomial Distribution

Figure 2. In this plot we present the number of times that our algorithms (Fair PA, Fair IID) and the baselines (SC, EHKS, DP) pick from each position of the input prophet problem stream. In (a) the stream consists of 50 sample from the uniform distribution and in (b) the stream consist of 1000 sample from the binomial distribution.

back. In order to achieve high quality feedback, we want to maximize the length of the call phone call duration while being fair with respect to the age of the interviewee. We divide the clients into 5 colors: under 30, 31-40, 41-50, 51-60, and more that 61 years old. For the sake of being fair, we let $\mathbf{p} = (1/5, 1/5, 1/5, 1/5, 1/5)$. In Figure 1 (c), we present the obtained results (along with the number of the records in the input for each color). Similar to the previous experiments, we observe that our algorithm and SCSA pick almost equal number of the times from each color while SA picks mostly (80% of the runs) from the forth color. Morevoer, we observe that our algorithm picks 1.347 times more elements than SCSA (+34.7%), and that it picks the maximum element of the color 1.760 times more often (+76.0%).

Influence maximization. We consider a dataset containing the influence of the users of the Pokec social network (Takac & Zábovský, 2012). The influence is computed as the number of the followers for each user. Selecting influencers has numerous applications, e.g., in advertising. In this experiment we want to be fair with respect to the body mass index (BMI) of the selected influencers. Therefore we divide the users into 5 colors according to their BMI: under weighted, normal, over weighted, obese type 1, and obese type 2. We let $\mathbf{p} = (1/5, 1/5, 1/5, 1/5, 1/5)$. The results are presented in Figure 1 (d).[3] Similar to the previous experiments our algorithm and SCSA picks almost equal number of each color while the Secretary algorithm picks only from two colors. Moreover, we observe that our algorithm picks 1.373 times more elements than SCSA (+37.3%), and picks the maximum element of the color 1.756 time more often than SCSA (+75.6%).

4.2. Prophet Experiments

In this section we evaluate our multi-color prophet algorithms (Algorithm 2 (Fair PA) and Algorithm 3 (Fair IID)). We focus on the case, where values are distributed i.i.d. and each candidate is a group on its own. We compare with the following baselines:

[3]For ease of representation, this experiment is ran 10^6 times.

- SC algorithm (Samuel-Cahn, 1984): This algorithm sets a single threshold so that the maximum is above this threshold with probability exactly $1/2$. It achieves an optimal 2-approximation for possibly non-identical independent distributions and arbitrary arrival order.
- EHKS algorithm (Ehsani et al., 2018): This algorithm sets a single threshold so that an individual candidate is accepted with probability $1/n$. It achieves an approximation of $(e + 1)/e \approx 1.58$ for possibly non-identical independent distributions and random arrival order.
- CFHOV algorithm (Correa et al., to appear): This algorithm sets a sequence of thresholds based on acceptance probabilities that result from solving a differential equation. It achieves an optimal 1.342-approximation for IID distributions.
- DP algorithm (e.g. Chow et al., 1971): This algorithm is the optimal threshold algorithm for the prophet problem, where thresholds are obtained by backward induction. This algorithm is optimal, even when distributions are different and candidates arrive in arbitrary order.

We consider two settings. In the first one the input stream consists of 50 samples from the uniform distribution in range [0, 1], and in the second one the input consists of 1000 samples from the binomial distribution with 1000 trials and $1/2$ probability of success of a single trial. For better comparability with existing algorithms, in both cases we assume each candidate is a group on its own. We run each algorithm $50,000$ times.

In Figure 2 we compare the number of times that our algorithms, SC, EHKS, and CFHOV pick from each position of the stream. The DP algorithm picks very unfairly and almost exclusively from the very end of the stream. As this would distort the readability of the figures we excluded this curve from the plots. We observe that both the SC and EHKS baselines pick candidates more from the first half of the stream compared to the second half (by more than a factor of 2), while DP picks mostly from the second half of the stream (by more than a factor of 4). So all these algorithms are unfair. In contrast, our algorithms select the same number of candidates throughout the stream. The average value of the chosen candidate for our Algorithm 2 (Fair PA), our Algorithm 3 (Fair IID), SC, EHKS, CFHOV, and DP for the uniform distribution is 0.501, 0.661, 0.499, 0.631, 0.752, 0.751, while for the binomial distribution it is 298.34, 389.24, 277.63, 363.97, 430.08, 513.34, respectively.

In conclusion, for both settings, both our algorithms Algorithm 2 and Algorithm 3 provide perfect fairness, while giving 66.71% and 88.01% (for the uniform case), and 58.12% and 75.82% (for the binomial case), of the value of the optimal, but unfair, online algorithm.

5. Conclusion and Open Problems

In this work, we explored questions of fairness and bias in natural multi-color variants of the two canonical problems of online selection, the secretary problem and the prophet problem. We designed optimal fair online algorithms for these problems, and validated the efficacy and fairness of these new algorithms on synthetic and real-world data.

As in many real-world settings the online decisions go beyond the single selection model studied here, there is ample opportunity for extending this line of work to combinatorial settings. We expect that building on the respective lines of work in the secretary, prophet and optimal stopping literature in general, could prove very fruitful.

Particularly exciting directions include an extension to matching problems (Kesselheim et al., 2013; Ezra et al., 2020; Gravin & Wang, 2019), allocation problems with matroid structure (Babaioff et al., 2018; Feldman et al., 2015b; Kleinberg & Weinberg, 2012; Dütting et al., 2020a), or even general combinatorial allocation problems (Feldman et al., 2015a; Dütting et al., 2020b).

Acknowledgments

José Correa was partially funded by ANID through grant FONDECYT 1190043. Andrés Cristi is supported by ANID under grant PFCHA/Doctorado Nacional/2018-21180347.

References

Babaioff, M., Immorlica, N., Kempe, D., and Kleinberg, R. Matroid secretary problems. *Journal of the ACM*, 65(6): 35:1–35:26, 2018.

Buchbinder, N., Jain, K., and Singh, M. Secretary problems via linear programming. *Mathematics of Operations Research*, 39(1):190–206, 2014.

Cayci, S., Gupta, S., and Eryilmaz, A. Group-fair online allocation in continuous time. In *Proceedings of the Annual Conference on Neural Information and Processing Systems (NeurIPS)*, 2020.

Celis, E., Keswani, V., Straszak, D., Deshpande, A., Kathuria, T., and Vishnoi, N. Fair and diverse DPP-based data summarization. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 716–725, 2018a.

Celis, L. E., Huang, L., and Vishnoi, N. K. Multiwinner voting with fairness constraints. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, (IJCAI)*, pp. 144–151, 2018b.

Celis, L. E., Straszak, D., and Vishnoi, N. K. Ranking with fairness constraints. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming, (ICALP)*, pp. 28:1–28:15, 2018c.

Chierichetti, F., Kumar, R., Lattanzi, S., and Vassilvitskii, S. Fair clustering through fairlets. In *Advances in Neural Information Processing Systems*, pp. 5029–5037, 2017.

Chierichetti, F., Kumar, R., Lattanzi, S., and Vassilvtiskii, S. Matroids, matchings, and fairness. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 2212–2220, 2019.

Chow, Y. S., Robbins, H. E., and Siegmund, D. *Great Expectations: The Theory of Optimal Stopping*. Houghton Mifflin, Boston, MA, USA, 1971.

Correa, J., Foncea, P., Hoeksma, R., Oosterwijk, T., and Vredeveld, T. Posted price mechanisms and optimal threshold strategies for random arrivals. *Mathematics of Operations Research*, to appear.

Dütting, P., Feldman, M., Kesselheim, T., and Lucier, B. Prophet inequalities made easy: Stochastic optimization by pricing nonstochastic inputs. *SIAM J. Comput.*, 49(3): 540–582, 2020a.

Dütting, P., Kesselheim, T., and Lucier, B. An o(log log m) prophet inequality for subadditive combinatorial auctions. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 306–317, 2020b.

Dwork, C., McSherry, F., Nissim, K., and A., S. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Theory of Cryptography Conference (TCC)*, pp. 265–284, 2006.

Dynkin, E. B. The optimum choice of the instant for stopping a markov process. *Soviet Mathematics Doklady*, 4: 627–629, 1963.

Ehsani, S., Hajiaghayi, M., Kesselheim, T., and Singla, S. Prophet secretary for combinatorial auctions and matroids. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 700–714, 2018.

Ezra, T., Feldman, M., Gravin, N., and Tang, Z. G. Online stochastic max-weight matching: Prophet inequality for vertex and edge arrival models. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, pp. 769–787, 2020.

Feldman, M. and Tennenholtz, M. Interviewing secretaries in parallel. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pp. 550–567, 2012.

Feldman, M., Gravin, N., and Lucier, B. Combinatorial auctions via posted prices. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 123–135, 2015a.

Feldman, M., Svensson, O., and Zenklusen, R. A simple O(log log(rank))-competitive algorithm for the matroid secretary problem. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1189–1201, 2015b.

Feldman, M., Svensson, O., and Zenklusen, R. Online contention resolution schemes. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1014–1033, 2016.

Ferguson, T. S. Who solved the secretary problem? *Statistical Science*, 4:282–289, 1989.

Gravin, N. and Wang, H. Prophet inequality for bipartite matching: Merits of being simple and non adaptive. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, pp. 93–109, 2019.

Gupta, S. and Salem, J. Closing the gap: Mitigating bias in online resume-filtering. In *Proceedings of the Conference on Web and Internet Economics (WINE)*, 2020.

Halabi, M. E., Mitrovic, S., Norouzi-Fard, A., Tardos, J., and Tarnawski, J. Fairness in streaming submodular maximization: Algorithms and hardness. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Kearns, M. and Roth, A. *The Ethical Algorithm: The Science of Socially Aware Algorithm Design*. Oxford University Press, 2019.

Kesselheim, T., Radke, K., Tönnis, A., and Vöcking, B. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *Proccedings of the European Symposium on Algorithms (ESA)*, pp. 589–600, 2013.

Kleinberg, R. and Weinberg, S. M. Matroid prophet inequalities. In *Proceedings of the ACM Symposium on Theory of Computing Conference (STOC)*, pp. 123–136, 2012.

Kumar, R., Lattanzi, S., Vassilvitskii, S., and Vattani, A. Hiring a secretary from a poset. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pp. 39–48, 2011.

Lindley, D. V. Dynamic programming and decision theory. *Applied Statistics*, 10:39–51, 1961.

Moro, S., Cortez, P., and Rita, P. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014.

Samuel-Cahn, E. Comparison of threshold stop rules and maximum for independent nonnegative random variables. *Annals of Probability*, 12:1213–1216, 1984.

Takac, L. and Zábovský, M. Data analysis in public social networks. In *Proceedings of the International Scientific Conference and International Workshop Present Day Trends of Innovations*, pp. 1–6, 2012.