Candidate Selections with Proportional Fairness Constraints

Xiaohui Bei
Nanyang Technological University
xhbei@ntu.edu.sg

Shengxin Liu*
Nanyang Technological University
sxliu@ntu.edu.sg

Chung Keung Poon
The Hang Seng University of Hong Kong
ckpoon@hsu.edu.hk

Hongao Wang
Nanyang Technological University
hongao.wang@ntu.edu.sg

ABSTRACT

Selecting a subset of candidates with various attributes under fairness constraints has been attracting considerable attention from the AI community, with applications ranging from school admissions to committee selections. The fairness constraints are usually captured by absolute upper bounds and/or lower bounds on the number of selected candidates in specific attributes. In many scenarios, however, the total number of selected candidates is not predetermined. It is, therefore, more natural to express these fairness constraints in terms of proportions of the final selection size. In this paper, we study the proportional candidate selection problem, where the goal is to select a subset of candidates with maximum cardinality while meeting certain proportional fairness constraints. We first analyze the computational complexity of the problem and show strong inapproximability results. Next, we investigate the algorithmic aspects of the problem in two directions. First, by treating the proportional fairness constraints as soft constraints, we devise two polynomial-time algorithms that could return (near) optimal solutions with bounded violations on each fairness constraint. Second, we design an exact algorithm with a fast running time in practice. Simulations based on both synthetic and publicly available data confirm the effectiveness and efficiency of our proposed algorithms.

KEYWORDS

Multi-winner selections; Proportional fairness constraints; Variable number of winners

ACM Reference Format:
Xiaohui Bei, Shengxin Liu, Chung Keung Poon, and Hongao Wang. 2020. Candidate Selections with Proportional Fairness Constraints. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 9 pages.

1 INTRODUCTION

The problem of selecting a collection of alternatives from a larger pool has a wide range of applications in the AI realm, ranging from qualified school admissions [1, 2, 10, 13, 14, 18, 19] to representative program committee selections [3, 7, 8, 11, 12, 20, 24]. To ensure sufficient representation of minorities, a number of recent research turn their attention to the issue of *fairness*. In the literature, the fairness constraints are usually defined based on attributes (or types) of candidates [7, 8]. Consider forming a program committee for an AI conference, for example. In order to ensure fairness among different sub-areas, one may want to diversify the selection and make sure that at least a certain number of senior members are selected in each sub-area. Formally speaking, consider a set of n candidates and a set of m characterization attributes such as expertise, gender, or region. Each candidate is associated with some attributes. Then the fairness constraints on the selected candidate set S require that given non-negative integers l_j and u_j, S should contain at least l_j and/or at most u_j candidates for each attribute $j \in [m]$.

However, in many scenarios, the fairness constraints expressed using *absolute* values are inadequate. Take school enrollment as an example [9]. In 1989, each school in the city of White Plains (New York) was required to have the same proportions of Blacks, Hispanics, and "Others" (which includes Whites and Asians). The plan allowed for a discrepancy among schools of only 5 percent. Translating this requirement to fairness upper bound constraints with absolute numbers, it (roughly) means that a school of capacity 1,000 can have at most about 350 students from each racial category. However, this translation relies on a critical assumption that the school will always be *fully* allocated, which is often not the case in real life. For example, the enrollment has dropped from 426,215 in 2000 to about 350,535 in 2013 observed by Chicago Public Schools, resulting in almost 50 percent of the schools being half-empty. If the school of capacity 1,000 ends up only enrolling 500 students, the absolute-valued fairness constraints may lead to a worst-case of 0 students enrolled in a specific category! This situation may render the fairness requirement null and void and lead to a clearly undesirable situation.

In this paper, we investigate this issue by looking at the *proportional candidate selection problem*, where the goal is to select a subset of candidates while satisfying certain *proportional fairness* constraints. Specifically, the proportional fairness constraints require that, for every attribute $j \in [m]$, the selected candidate set S has at least α_j fraction and at most β_j fraction of candidates that possess this attribute. Note that the proportional constraints can be trivially satisfied by selecting an empty set. Instead, we are interested in finding a feasible set of *maximum size* under an overall capacity constraint.

1.1 Our Contributions

We present both hardness and algorithmic results for the proportional candidate selection problem in this paper. In Section 3, we first consider the computational complexity of the problem. Our

*Corresponding author.

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

first result is a hardness result on finding even a *non-empty* feasible solution.

Theorem 1.1 (NP-hardness of feasibility). *It is NP-hard to check whether there is a* nonempty *feasible solution of the proportional candidate selection problem.*

Consequently, this leads to the following strong inapproximability result of the optimization problem.

Corollary 1.2 (Inapproximability). *The proportional candidate selection problem is NP-hard to approximate within any ratio* $\gamma \geq 1$.

Given the intractability of the proportional candidate selection problem, it naturally leads to two algorithmic questions:

- Can we devise a polynomial time algorithm if we treat the fairness constraints *soft* (e.g., each proportional fairness constraint can be violated by a small additive/multiplicative factor)?
- Can we design time-efficient algorithm that runs fast in practice even though the worst-case running time remains exponential?

In this paper we answer both questions in the affirmative, and our proposed algorithms are verified through various experiments in Section 5.

Algorithms with Soft Constraints. We first give a randomized rounding algorithm in Section 4.1:

Theorem 1.3 (Informal). *Under mild conditions, for some* $0 < \epsilon < 1$*, with positive probability our randomized algorithm produces a solution with size at least* $1-\epsilon$ *fraction of the optimum size while each constraint is violated by at most a multiplicative ratio of* $(1+\epsilon)/(1-\epsilon)$.

The algorithm utilizes randomized rounding of the linear programming relaxation, and the approximation is guaranteed by the concentration property using Chernoff bounds.

Then, in Section 4.2 we give another polynomial time algorithm that returns a solution with bounded additive violations on all fairness constraints. More specifically, we have the following:

Theorem 1.4 (Informal). *Our deterministic algorithm produces a solution with size at least the optimum solution while each constraint is violated by at most an additive error of* $2\Delta + 1$*, where* Δ *is the maximum number of attributes that a candidate can have.*

The algorithm makes use of the iterative method (see [21]) developed for solving many combinatorial optimization problems.

Exact Solution. In the second direction, we focus on exact algorithms to the proportional candidate selection problem in Section 4.3. A commonly used method in practice is to formulate the NP-hard problem as an Integer Linear Programming (ILP) and directly apply an ILP solver to solve it. While in our algorithm, we first enumerate the possible value of the optimum solution, and solve a "constant" version of the problem for each of our guess. Intuitively, this straightforward decomposition can reduce the number of nonzero coefficients in the ILP, which leads to faster implementation in practice. Moreover, we propose an iterative framework that guesses the optimum solution in a more efficient way, leading to fewer iterations of solving the "constant" versions.

1.2 Related Works

There is a growing literature in computational social choice on the fairness (or distributional) issue. In particular, some previous works considered the assignment and matching mechanisms subject to either lower- or upper-bound constraints on different types of objects [1, 2, 4–6, 10, 13, 14, 18, 19, 22, 23]. Most of these works made use of absolute values in their lower- and/or upper-bound covariants, which have distinct difference with our model where proportional fairness constraints are imposed. The only exception is the work by Nguyen and Vohra [22] where their model only deals with the special case that each candidate only has *one* attribute. Our model is more general in the sense that we allow each candidate to possess multiple attributes.

Another related line of research on fairness (or diversity) constraints is on the multiwinner voting problem [3, 7, 8, 20]. In the standard setting of the multiwinner voting problem, the winning sets are required to have *exactly* a fixed number of k candidates. Thus it is natural to use absolute values, instead of proportions, to capture the fairness constraints. Some researchers also noticed that the number of winners could be variable in some real-world scenarios [12, 16, 24]. However, none of these works addressed the proportional fairness constraints.

2 PRELIMINARIES

Consider a set C of n candidates and a set P of m properties (or attributes) where each candidate i possesses a set of properties $P_i \subseteq P$. Moreover, let Δ be the maximum number of properties that a candidate can possess, i.e., $\Delta = \max_{i \in C} |P_i|$. We use "attributes" and "properties" interchangeably in this paper.

We first define the proportional fairness constraints. Denote $\vec{\alpha} = \{\alpha_1, \ldots, \alpha_m\}$ and $\vec{\beta} = \{\beta_1, \ldots, \beta_m\}$ where $0 \leq \alpha_j \leq \beta_j \leq 1$ for all $j \in [m]$.

Definition 2.1. Given candidate set C and $\vec{\alpha}, \vec{\beta}$, a subset $C' \subseteq C$ of candidates is said to satisfy the *proportional fairness constraints* with $\vec{\alpha}$ and $\vec{\beta}$ if for each property $j \in P$, the number of candidates in C' that have property j is at least $\alpha_j |C'|$ and at most $\beta_j |C'|$, i.e.,

$$\alpha_j |C'| \leq |\{i \in C' \mid j \in P_i\}| \leq \beta_j |C'|.$$

Now we are ready to define our main problem.

Definition 2.2. Given fairness parameters $\vec{\alpha}, \vec{\beta}$ and cardinality threshold k, the *Proportional Candidate Selection Problem* aims to find a subset of candidates C' of *maximum size*, such that $|C'| \leq k$ and C' satisfies all proportional fairness constraints. We denote $|C'| \leq k$ as the *cardinality constraint*.

Integer Linear Programming formulation. Let $p_{ij} = 1$ if candidate i has attribute j and $p_{ij} = 0$ otherwise. The proportional candidate selection problem can be easily formulated as the following ILP:

$$\max. \sum_{i \in C} x_i \tag{1}$$

$$\text{s. t. } \alpha_j \sum_{i \in C} x_i \leq \sum_{i \in C} p_{ij} x_i \leq \beta_j \sum_{i \in C} x_i \qquad \forall j \in P, \tag{2}$$

$$\sum_{i \in C} x_i \leq k \tag{3}$$

$$x_i \in \{0, 1\} \qquad \forall i \in C. \tag{4}$$

Here x_i is a binary variable that represents whether or not candidate i is selected in the solution. The natural linear relaxation of ILP (1-4), which is denoted by LP (1-4), is to replace $x_i = \{0, 1\}$ with $x_i \in [0, 1]$ for each element $i \in C$ in Constraint (4). We also denote OPT_{1-4} and LP_{1-4} as optimum values of ILP (1-4) and LP (1-4), respectively. It is clear that $\mathsf{LP}_{1-4} \geq \mathsf{OPT}_{1-4}$. Similar notations can be defined analogously for other ILPs.

3 HARDNESS RESULTS

We investigate the computational complexity of the proportional candidate selection problem and show hardness results in this section.

Given the NP-hardness of the general candidate selection problem with absolute-valued constraints, it is not surprising that the proportional candidate selection problem is also NP-hard. In the following we show an even stronger claim: We prove that it is NP-hard even to decide whether there exists a nonempty feasible solution.

Theorem 1.1. *It is NP-hard to check whether there is a* nonempty *feasible solution of the proportional candidate selection problem.*

Our proof idea is as follows. We will construct a problem instance, in which all the feasible solutions are restricted to have value either 0 or a specific nonzero number (say, k'). Then we will show that, even knowing the value of k', it is NP-hard to decide whether there exists a feasible solution with value k' for this instance.

We are ready to prove Theorem 1.1:

Proof of Theorem 1.1. We reduce the NP-hard problem of *Exact Cover by 3-Sets* (X3C) [15] to our proportional candidate selection problem. Given a set F of $3k_1$ elements and a collection T of k_2 triples (i.e., three-element subsets of F), the X3C problem asks whether there exists a sub-collection T' of T with size k_1 such that every element in F appears in *exactly* one triple in T'.

Given an X3C instance, we construct our proportional candidate selection problem instance as follows. For each element and each triple in the X3C problem, we have a corresponding property j in P and a corresponding candidate i that has exactly 3 properties in C. We additionally have a special candidate i^* and a special property j^* where i^* only has a single property j^* and j^* is only possessed by a single candidate i^*. Thus we have $|P| = 3k_1 + 1$ and $|C| = k_2 + 1$. Let $\alpha_j = \beta_j = 1/(k_1 + 1)$ for all $j \in P$. This implies that we require each property is contained in *exactly* $1/(k_1 + 1)$ fraction of the size of selected candidates (i.e., the inequalities in Constraint (2) become equalities for all properties).

Given the construction, we show that every nonempty solution to the proportional candidate selection problem must

(1) contain the candidate i^* with property j^*; and
(2) have value of $k_1 + 1$.

The reason is as follows. Let C' be a feasible solution of size $k' > 0$. The correctness of part (1) can be easily seen as candidate i^* is the only candidate with property j^* and $\alpha_{j^*} = 1/(k_1 + 1) > 0$.

Next we prove part (2) using the counting argument. We focus on the properties in $P \setminus \{j^*\}$. Since each candidate except i^* has exactly 3 properties by construction and $i^* \in C'$ by part (1), all candidates in $C' \setminus \{i^*\}$ will have $3(|C'| - 1) = 3(k' - 1)$ properties (counting multiplicity) in total. On the other hand, since $\alpha_j = \beta_j = 1/(k_1 + 1)$ for all $j \in P$, every property in $P \setminus \{j^*\}$ must be possessed by exactly $k'/(k_1 + 1)$ candidates in $C' \setminus \{i^*\}$. Therefore the total number of properties (counting multiplicity) possessed by all candidates in $C' \setminus \{i^*\}$ is $(|P| - 1) \cdot k'/(k_1 + 1) = 3k_1 \cdot k'/(k_1 + 1)$. Thus we have $3(k' - 1) = 3k_1 \cdot k'/(k_1 + 1)$, which implies that $k' = k_1 + 1$.

Since each property in this feasible solution is possessed by exactly $k'/(k_1 + 1) = 1$ candidate in C'. A nonempty solution C' to the proportional candidate selection problem thus implies an exact cover $C' \setminus \{i^*\}$ of X3C.

On the other hand, given an exact cover T' for an instance of X3C, it is easy to see that $T' \cup \{i^*\}$ is a feasible solution to the corresponding instance of the proportional candidate selection problem. This concludes the reduction and the proof of this theorem. □

Furthermore, we have the following inapproximability result for proportional candidate selection problem, which can be directly derived from Theorem 1.1.

Corollary 1.2 (Inapproximability). *The proportional candidate selection problem is NP-hard to approximate within any ratio $\gamma \geq 1$.*

Proof. Suppose that we have a γ-approximate algorithm $\mathcal{A}$ for some $\gamma \geq 1$. It is easy to see that we can utilize $\mathcal{A}$ to distinguish whether there exists a nonempty feasible solution for any instance: Report 'yes' if $\mathcal{A}$ returns a nonempty solution and 'no' otherwise. This contradicts Theorem 1.1 and completes our proof. □

FPT-Algorithm with respect to m. We remark that the number of properties m is a variable in the hardness proof of Theorem 1.1. When m is fixed, the proportional candidate selection problem admits a fixed-parameter tractable (FPT) algorithm with respect to m. Specifically, given a problem instance, we can guess the optimum value from n to 0, and with each guess transform the problem into another problem with constant fairness constraints, i.e., lower- and upper-bound covariants that correspond to fairness constraints are absolute numbers. For this "constant" version of the problem, Bredereck et al. [7] (Theorem 10 in their work) showed an FPT-algorithm with respect to m via solving a mixed ILP with 2^m integer variables. However, the algorithm has m as the exponent in its time complexity. With m as large as 6 in our experiments in Section 5, the algorithm cannot terminate in reasonable time for most instances. Thus we do not find such an FPT-algorithm applicable in real-world scenarios and view it more of theoretical interest.

4 ALGORITHMS

In this section we present our algorithmic results. We devise two polynomial time approximation algorithms in Sections 4.1 and 4.2, at the expense of having bounded multiplicative and additive violations on proportional fairness constraints respectively. Then Section 4.3 gives an algorithm based on the "guess-and-verify" strategy to solve the problem exactly.

4.1 Randomized Algorithm

Our randomized algorithm RandRounding follows the classic methodology of randomized rounding. In particular, the algorithm consists of two major steps. In the first step, RandRounding solves

LP (1-4) and obtains an optimal fractional solution x^* with optimum value LP_{1-4}. The second step applies standard randomized rounding to round x^* to an integral solution $\hat{x}$, i.e., for each i, we set $\hat{x}_i = 1$ with probability x_i^*; 0 with probability $1 - x_i^*$. Let the size of $\hat{x}$ be ALG. For convenience, we use LP and OPT to denote LP_{1-4} and OPT_{1-4} in this subsection, respectively.

In the following we show that the rounded integral solution $\hat{x}$ is close to the actual optimal solution with high probability. We restate the result of Theorem 1.3 in a formal way:

Theorem 4.1. *For any $\sqrt{3\ln(2m+3)/\min_j \alpha_j \mathsf{OPT}} \le \epsilon < 1$, with positive probability, the output of Algorithm* RandRounding *satisfies*

(1) $\mathsf{ALG} \ge (1-\epsilon)\mathsf{OPT}$,
(2) *the cardinality constraint $|C'| \le k$ is violated by a multiplicative ratio of no more than $(1+\epsilon)$, and*
(3) *each proportional fairness constraint is violated by a multiplicative ratio of no more than $(1+\epsilon)/(1-\epsilon)$, i.e., $\alpha_j \frac{1-\epsilon}{1+\epsilon}\mathsf{ALG} \le \sum_i p_{ij}\hat{x}_i \le \beta_j \frac{1+\epsilon}{1-\epsilon}\mathsf{ALG}$ for all $j \in P$.*

Theorem 4.1 can be easily proved by applying Chernoff bounds where the details are omitted due to the page limit.

We note that in real-world applications, the violation factor ϵ is usually rather small, since there are usually very few number of properties to consider, i.e. m is usually a very small number, and the the number of candidates (and potentially the value of OPT) is often relatively large.

4.2 Iterative Algorithm

We now present another algorithm, denoted by Iterative, with soft constraints which returns a solution with bounded additive violations on each fairness constraints. We restate the result of Theorem 1.4 in a formal way:

Theorem 4.2. *Algorithm* Iterative *returns a solution with value* ALG *for the proportional candidate selection problem such that*

(1) $\mathsf{ALG} \ge \mathsf{OPT}_{1-4}$, *and*
(2) *each proportional fairness constraint is violated by an additive factor of no more than $2\Delta + 1$.*

The general idea of our approach is as follows. We first transform ILP (1-4) to another ILP formulation and show that a "good" solution to the new formulation is also a "good" one to ILP (1-4). Then we solve the new ILP formulation using the iterative method [17, 21].

Step 1: Transformation. Let x^* be a solution that corresponds to LP_{1-4}. We also let $f_j = \lfloor \alpha_j \sum_{i\in C} x_i^* \rfloor$ and $g_j = \lceil \beta_j \sum_{i\in C} x_i^* \rceil$ for each $j \in P$, and let $f_0 = \lfloor \sum_{i\in C} x_i^* \rfloor$ and $g_0 = \lceil \sum_{i\in C} x_i^* \rceil$.

$$\max. \sum_{i\in C} x_i \tag{5}$$

$$\text{s. t. } f_j \le \sum_{i\in C} p_{ij}x_i \le g_j \qquad \forall j \in P, \tag{6}$$

$$f_0 \le \sum_{i\in C} x_i \le g_0 \tag{7}$$

$$x_i \in \{0,1\} \qquad \forall i \in C. \tag{8}$$

Note that in this transformed ILP, we replace the cardinality constraint, i.e., Constraint (3), in ILP (1-4) with Constraint (12) which will be useful for our analysis later.

We next show a relation between ILP (1-4) and ILP (10-13).

Lemma 4.3. *Suppose we have an algorithm $\mathcal{A}$ that returns an integral solution y for ILP (10-13) such that:*

- $\sum_{i\in C} y_i \ge \mathsf{LP}_{10-13}$,
- $f_0 \le \sum_{i\in C} y_i \le g_0$, *and*
- *there is an additive violation of at most s on the fairness constraints (Constraint (11)), i.e.,*

$$f_j - s \le \sum_{i\in C} p_{ij}y_i \le g_j + s \ \forall j \in P.$$

Then the integral solution y produced by $\mathcal{A}$ is a solution to ILP (1-4) satisfying:

(1) $\sum_{i\in C} y_i \ge \mathsf{OPT}_{1-4}$,
(2) $\lfloor \sum_{i\in C} x_i^* \rfloor \le \sum_{i\in C} y_i \le \lceil \sum_{i\in C} x_i^* \rceil \le k$, *and*
(3) *there is an additive violation of at most $s+2$ on the fairness constraints (Constraint (2)), i.e.,*

$$\alpha_j \sum_{i\in C} y_i - s - 2 \le \sum_{i\in C} p_{ij}y_i \le \beta_j \sum_{i\in C} y_i + s + 2 \ \forall j \in P.$$

Proof. Part 1 is straightforward since it is obvious to see that x^* (an optimal solution to LP_{1-4}) is a feasible solution for LP_{10-13}. Part 2 is a direct consequence of the property of solution x^* and Algorithm $\mathcal{A}$.

We focus on proving Part 3. Fix a property $j \in P$. By Algorithm $\mathcal{A}$, we have

$$\begin{aligned}\sum_{i\in C} p_{ij}y_i &\le \lceil \beta_j \sum_{i\in C} x_i^* \rceil + s \le \beta_j \sum_{i\in C} x_i^* + s + 1 \\ &\le \beta_j(\lfloor \sum_{i\in C} x_i^* \rfloor + 1) + s + 1 \le \beta_j \lfloor \sum_{i\in C} x_i^* \rfloor + s + 2 \\ &\le \beta_j \sum_{i\in C} y_i + s + 2,\end{aligned}$$

where the fourth inequality follows as $\beta_j \le 1$ and the last inequality is due to the property of Algorithm $\mathcal{A}$. Similarly, we also have $\sum_{i\in C} p_{ij}y_i \ge \alpha_j \sum_{i\in C} y_i - s - 2$, which completes the proof of Lemma 4.3. □

Step 2: Iterative Method. Guided by the relation explored in Lemma 4.3, we now focus on solving ILP (10-13). Before showing the algorithm, we first give a characterization of extreme point solutions of LP (10-13). We next present an important lemma which is the core of the iterative method [21]:

Lemma 4.4 (Rank Lemma [21]). *Let $\mathcal{P} = \{x \mid Ax \ge b, x \ge 0\}$ and let x be an extreme point solution of $\mathcal{P}$ such that $x_i > 0$ for each i. Then any maximal number of linearly independent tight constraints of the form $A_i x = b_i$ for some row i of A equals the number of variables.*

Let $j^\forall$ be a special property such that all candidates have this property $j^\forall$, where the corresponding constraint is shown as Constraint (12). The following lemma is then a direct application of the Rank Lemma (Lemma 4.4) which gives the characterization of extreme point solutions of LP (10-13):

Lemma 4.5. *For any extreme point solution x to LP (10-13) with $0 < x_i < 1$ for each $i \in C$, there exists $W \subseteq P \cup \{j^\forall\}$ such that*

(1) *each constraint that corresponds to W is tight, i.e.,*
   - *if $j \in W \cap P$, $\sum_{i\in C} p_{ij}x_i$ equals either f_j or g_j;*
   - *otherwise, i.e., $j = j^\forall$, $\sum_{i\in C} x_i$ equals either f_0 or g_0.*

Algorithm 1 Iterative

1: Initialize $y_i \leftarrow 0$ for all i.
2: **while** the current LP (10-13) is not empty **do**
3: Find an extreme point optimal solution x for LP (10-13).
4: For each candidate i with $x_i = 0$, delete i from C. Update LP (10-13).
5: For each candidate i with $x_i = 1$, delete i from C, set $y_i \leftarrow 1$, and decrease f_0, g_0, and f_j and g_j for each property $j \in P_i$ by 1, respectively. Update LP (10-13).
6: For each property j, delete j from P if the current number of candidates that have property j is at most $2\Delta - 1$, i.e., $|\{i \in C \mid j \in P_i\}| \leq 2\Delta - 1$. Update LP (10-13).
7: **return** y

(2) *The constraints corresponding to W are linearly independent.*
(3) $|W| = |C|$.

Now we are ready to present Iterative (Algorithm 1) where the corresponding result is shown in Theorem 4.6.

Theorem 4.6. *Algorithm* Iterative *returns a solution y for ILP (10-13) such that:*

- $\sum_{i \in C} y_i \geq \mathsf{LP}_{10-13}$,
- $\lfloor \sum_{i \in C} x_i^* \rfloor \leq \sum_{i \in C} y_i \leq \lceil \sum_{i \in C} x_i^* \rceil$, *and*
- *there is an additive violation of at most $2\Delta - 1$ on the fairness constraints (Constraint (11)), i.e., $\forall j \in p$,*

$$\lfloor \alpha_j \sum_{i \in C} x_i^* \rfloor - 2\Delta + 1 \leq \sum_{i \in C} p_{ij} y_i \leq \lceil \beta_j \sum_{i \in C} x_i^* \rceil + 2\Delta - 1.$$

Proof. Algorithm Iterative processes in iterations where we get a (strictly) smaller LP after each iteration. In the following we will show that in each iteration, either we can set some variables to 0 or 1 in the original LP, or at least one constraint can be removed.

First we show that this theorem holds if Iterative terminates successfully. First, observe that we update the linear program in Steps 4-5 according to whether $x_i = 0$ or 1 such that the residual linear programming solution (current LP solution restricted to those x_i's with value in the range of $(0, 1)$) remains a feasible solution for the modified linear program in the next iteration. This implies that the size of the current solution y plus the size of the LP solution, i.e., $\sum_{i \in C} y_i + x_i$ is always feasible with respect to f_0 and g_0 in the original LP during the algorithm. Also, in Step 6 when we remove a fairness constraint, the current LP solution remains a feasible solution. Therefore the size of the current solution y, i.e., $\sum_{i \in C} y_i$ plus the size of the LP solution does not decrease in any iteration, so at the final step the size of y is at least the cost of the first LP solution, which is at least LP_{10-13}. Moreover, since we only remove a fairness constraint of a property when it is possessed by at most $2\Delta - 1$ candidates, the fairness constraints are violated by at most $2\Delta - 1$.

Thus it remains to show that Algorithm 1 always terminates successfully. That is, it can always either find a candidate i with $x_i = 0$ in Step 4 or $x_i = 1$ in Step 5, or finds a property j such that there are at most $2\Delta - 1$ candidates with property j in the current candidate set, i.e., $|\{i \in C \mid j \in P_i\}| \leq 2\Delta - 1$, in Step 6.

Suppose by contradiction that none of the above conditions holds. Then we have $0 < x_i < 1$ for each $i \in C$ and $|\{i \in C \mid j \in P_i\}| \geq 2\Delta$ for each $j \in P$. We show the contradiction via a counting argument. We assign 2Δ tokens to each candidate $i \in C$ for a total of $2\Delta|C|$ tokens. For each candidate i, we redistribute one token to each property $j \in P_i$, and Δ tokens to property $j^\forall$. This can be done because each candidate has at most Δ properties. Next, we will show that the constraints in W can collect $2\Delta|W|$ tokens in total while there are still some tokens left. This would imply $|C| > |W|$ which contradicts to Lemma 4.5.

For each constraint $j \in W \cap P$, it collects at least 2Δ tokens since $|\{i \in C \mid j \in P_i\}| \geq 2\Delta$. On the other hand, property $j^\forall$ collects $\Delta|C|$ tokens. We then consider two cases:
Case (1). $W \subseteq P$. In this case, we know that W collects at least $2\Delta|W|$ tokens and property $j^\forall$ collects $\Delta|C|$ tokens. Since we have at most $2\Delta|C|$ tokens in total, this contradicts Part 3 of Lemma 4.5.
Case (2). $|W \cap P| = |W| - 1$. We know that $W \cap P$ collects at least $2\Delta|W \cap P| = 2\Delta(|W| - 1)$ tokens and property $j^\forall$ collects $\Delta|C|$ tokens.

- If $|C| > 2$, this contradicts to Part 3 of Lemma 4.5.
- If $|C| < 2$, we note that $|C| \neq 0$ since the current LP (10-13) is not empty as in line 2. Hence $|C| = 1$. By assumption we have $0 < x_i < 1$ for each $i \in C$, and f_0, g_0 are integers, this contradicts Part 1 of Lemma 4.5 where $\sum_{i \in C} x_i$ should be equal to either f_0 or g_0.
- If $|C| = 2$, this means that each candidate in C has exactly Δ properties since otherwise we already have the contradiction to Part 3 of Lemma 4.5. Then it is easy to see that for each i, we have $\sum_{j \in W \cap P} p_{ij} = \Delta \cdot 1$ which shows linear dependence to Constraint (12). Hence, we have the desired contradiction to Part 2 of Lemma 4.5.

□

By Theorem 4.6 and Lemma 4.3, we complete the proof of Theorem 4.2.

Note that both RandRounding and Iterative consider soft constraints and may select more candidates than the cardinality constraint. This is a feasible assumption in many applications, e.g., the school admission, which could tolerate a small violation on the total number of selected candidates. When the application requires hard cardinality constraint, we can randomly remove a few candidates from the selected set to meet the requirement. Because the violation is guaranteed to be small, these removals will not affect other constraints significantly.

4.3 Exact Solution

Besides algorithms with soft constraints, we also investigate exponential-time exact algorithms that run fast in practice.

We first consider an equivalent ILP formulation (see ILP (14-18)) with only upper-bound fairness constraints. The transformation is shown as follows. For each property j, we introduce a corresponding property j' that is possessed by those candidates that do not have j (i.e., $j' \in P_i$ if and only if $j \notin P_i$ for each $i \in C$). We collect these new property j''s as P' and set $\beta_{j'} = 1 - \alpha_j$. For example, a requirement that "at least 40 percent of selected candidates are female" can be translated to "at most 60 percent of students are non-female", where *non-female* is a new attribute introduced. It is easy to check that

Algorithm 2 GuessAndVerify

Input: an instance of proportional candidate selection
Output: a solution of value ALG

1: Set $s_1 \leftarrow \lfloor \mathsf{LP} \rfloor$ be an initial upper bound on OPT.
2: Set $p \leftarrow 1$.
3: **while true do**
4: Solve the ILP (19-23) with $s = s_p$, and denote the optimum value and solution by y_p and x^p.
5: **if** $y_p \geq s_p$ **then**
6: **return** x^p
7: **else**
8: Set $s_{p+1} \leftarrow y_p$.
9: Set $p \leftarrow p + 1$.

ILP (14-18) with only upper-bound constraints is equivalent to the original one (ILP (1-4)).

$$\max. \sum_{i \in C} x_i \tag{9}$$

$$\text{s. t.} \sum_{i \in C} p_{ij} x_i \leq \beta_j \sum_{i \in C} x_i \qquad \forall j \in P, \tag{10}$$

$$\sum_{i \in C} \overline{p}_{ij} x_i \leq (1 - \alpha_j) \sum_{i \in C} x_i \qquad \forall j \in \overline{P}, \tag{11}$$

$$\sum_{i \in C} x_i \leq k \tag{12}$$

$$x_i \in \{0, 1\} \qquad \forall i \in C. \tag{13}$$

Here $\overline{p}_{ij} = 1$ if candidate i does not have attribute j, i.e., $p_{ij} = 0$, and $\overline{p}_{ij} = 0$ otherwise.

For our exact algorithm, we make use of a "guess-and-verify" strategy, in which the following ILP is solved for different values of guess s:

$$\max. \sum_{i \in C} x_i \tag{14}$$

$$\text{s. t.} \sum_{i \in C} p_{ij} x_i \leq \beta_j \cdot s \qquad \forall j \in P, \tag{15}$$

$$\sum_{i \in C} \overline{p}_{ij} x_i \leq (1 - \alpha_j) \cdot s \qquad \forall j \in \overline{P}, \tag{16}$$

$$\sum_{i \in C} x_i \leq k \tag{17}$$

$$x_i \in \{0, 1\} \qquad \forall i \in C. \tag{18}$$

For convenience, we use LP and OPT to denote LP_{1-4} and OPT_{1-4} below, respectively. The algorithm goes in iterations (lines 3-9) as follows. For the p-th iteration, we first *guess* an upper bound s_p on OPT. To guarantee s_1 is an initial upper bound, we set $s_1 \leftarrow \lfloor \mathsf{LP} \rfloor$ in line 1 since we know $\lfloor \mathsf{LP} \rfloor \geq \mathsf{OPT}$. Then we issue an ILP (19-23) with $s = s_p$ to obtain the optimum solution y_p (line 4). We next consider the value of y_p. If $y_p \geq s_p$ in lines 5-6, we immediately return x_p as the optimum solution ALG. Otherwise, i.e., $0 \leq y_p < s_p$, in lines 7-9 we derive another guess $s_{p+1} \leftarrow y_p$ for the $(p+1)$-st iteration. Note that the case of $y_p \geq s_p$ in line 5 coincides with case one of $y_p = s_p$, i.e., the case of $y_p > s_p$ will not happen.

We note that one cannot simply adopt a binary search approach for finding OPT, because the solution space is not monotone. That is, a feasible solution with value s does not imply there exists feasible solution with value $s' < s$.

We now claim the while loop of Algorithm 2 is processed in finite number of iterations:

Lemma 4.7. *The while loop (lines 3-9 of Algorithm 2) will terminate in $O(n)$ iterations.*

Proof. Suppose that the while loop terminates after the t-th iteration, i.e., ALG $= y_t$. In other words, t is the first iteration that satisfies $y_t \geq s_t$ in lines 5-6. For all $1 < p < t$, we will prove that $s_{p-1} > s_p$.

Consider a fixed iteration p such that $1 < p < t$. Assume, to the contrary, that $s_{p-1} \leq s_p$. We have $s_p = y_{p-1}$ by line 8 of Algorithm GuessAndVerify. Thus $s_{p-1} \leq y_{p-1}$, which means that the algorithm should terminate in the $(p-1)$-st iteration due to lines 5-6, a contradiction. Thus we know the sequence of $\{s_i\}$ is strictly decreasing.

We also know that there exists a trivial solution with value 0 of ILP (14-18). Suppose the q-th guess is $s_q = 0$, then we have $y_q = s_q = 0$ which implies $s_t \geq 0$. This also means that Algorithm 2 can always return a feasible solution. Moreover, we have $s_1 = k$, where $k = O(n)$, as shown in line 1. Thus we complete our proof. □

We next prove the correctness of Algorithm GuessAndVerify.

Lemma 4.8. *Algorithm* GuessAndVerify *solves the proportional candidate selection problem correctly.*

Proof. Let s_t be the first guess such that $s_t = y_t$. In other words, Algorithm 2 returns y_t as the optimum solution ALG in line 6. We will show $y_t = \mathsf{OPT}$ in the following.

Suppose that $y_t > \mathsf{OPT}$. Since $s_t = y_t$, we know that y_t corresponds to a feasible solution for the problem which contradicts the optimality of OPT.

Suppose that $y_t < \mathsf{OPT}$. Note that this case cannot happen with $\mathsf{OPT} = 0$ since $y_t \geq 0$. From the proof of Lemma 4.7, we have $s_1 > s_2 > \cdots > s_{t-1} > s_t \geq 0$. Thus there must exist a guess s_j such that $s_j \geq \mathsf{OPT} > s_{j+1} \geq s_t$ where $1 \leq j \leq t-1$. We observe that a larger guess implies a larger optimum value for ILP (19-23), i.e., $s_a \geq s_b$ implies $y_a \geq y_b$. Since $s_j \geq \mathsf{OPT}$, we have $y_j \geq y'_{opt}$ where y'_{opt} is the optimum value of ILP (19-23) with guess OPT. Then we conclude that $y'_{opt} \geq \mathsf{OPT}$ since, otherwise, it would violate the fact that OPT is a feasible (and optimum) solution to the proportional candidate selection problem. Moreover, according to our algorithm, we have $s_{j+1} = y_j$. Thus, $s_{j+1} = y_j \geq y'_{opt} \geq \mathsf{OPT}$ which contradicts the assumption that $\mathsf{OPT} > s_{j+1}$. The lemma follows. □

We remark that this framework is capable of solving the proportional candidate selection problem as long as we have an exact solution, either in polynomial or exponential time, to solve ILP (19-23) at hand.

5 EXPERIMENTS

In this section we conduct empirical evaluations of our proposed algorithms. Our experiments are partitioned into two parts based on

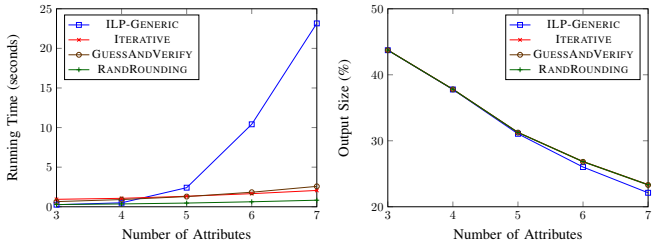

Figure 1: Running time and output size (%) with different number of attributes, respectively.

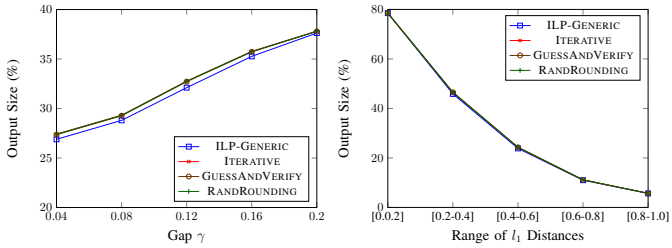

Figure 2: Output size (%) with different gap γ and ℓ_1 distances respectively.

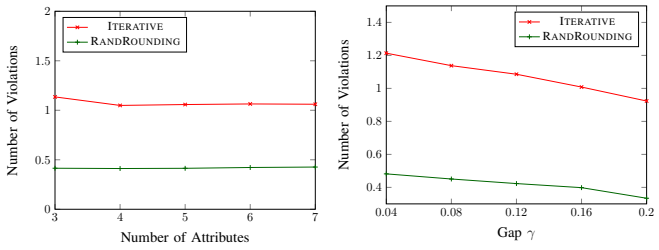

Figure 3: Violation with different number of attributes and gap γ respectively.

the data generation settings: the first one aims to test the sensitivities of our algorithms under different parameter settings through the randomly generated data, and the other evaluates the performance of our algorithms via a case study based on the real-world data.

Experiment setups. We compare RandRounding (Section 4.1), Iterative (Section 4.2), and GuessAndVerify (Section 4.3) with the baseline method ILP-Generic, i.e., the ILP (1-4) is solved by a standard ILP solver directly. All experiments ran on a computer with Intel Xeon E5-2630v4 @2.2 GHz CPU and 64 GB RAM. Programs were coded in Python 3.7, and all the ILPs and LPs were solved by the Gurobi Optimizer (version 8.1.0). For all the tests, we perform 100 repeated runs per experiment and show the averages.

5.1 Sensitivity Analysis over Synthetic Data

First we evaluate the above four algorithms using the synthetic data. In this experiment, for each attribute j, we first randomly generate a probability p_j and assign each candidate with this attribute with probability p_j independently. Then for the proportional constraints, for each attribute j we randomly generate a pair of parameters α_j and β_j such that their *gap* $\gamma = \alpha_j - \beta_j$ is fixed. The cardinality threshold is set as the total number of candidates. That is, we do not consider the cardinality constraint in these experiments. This is because: (1) it simplifies the experiments as we already have several dimensions of the parameters to consider; (2) according to our previous study, determining a suitable value of cardinality threshold k is dataset-specific: when k is too small, the number of selected candidates always reaches k; when k is too large, imposing the cardinality constraint does not affect the current result.

We test our algorithms on a number of different problem instances generated by varying the number of candidates, the number of attributes, as well as the gap γ between α_j and β_j for each attribute j.

Experiment results. Figure 1 shows the running times and output sizes in percentage (i.e., the output size over the problem size) of different algorithms when the number of attributes varies from 3 to 7. One can see clear trends that when the number of attributes increases, the running time increases while the output size decreases. This is because the problem becomes harder to solve if we have more attributes to deal with.

The left part of Figure 2 shows how the output size in percentage (i.e., the output size over the problem size) of the algorithm changes with regard to the gap γ. When γ increases, the fairness constraints become easier to satisfy, which is reflected by the larger output size of all algorithms.

We also plot the output size of our algorithms with regard to the ℓ_1-distance between p_j and center of $[\alpha_j, \beta_j]$, shown in the right part of Figure 2. One can see that when this distance grows larger, all algorithms can only select a significantly smaller fraction of the candidates into the solution set.

Note that in Figures 1 and 2, the (average) output size of ILP-Generic is smaller than that of GuessAndVerify. It is because in the synthetic dataset, there exist "hard" instances for which ILP-Generic does not return any (non-trivial) feasible solution within the cut-off time (set as 30 seconds in our experiments). In such situation, we simply use 0 (which is a trivial feasible solution) as the size of the selected set, which clearly makes the average output size smaller. On the other hand, our GuessAndVerify can finish all instances within the cut-off time of 30 seconds.

Finally, because RandRounding and Iterative are approximation algorithms, their outputs may have violations over the fairness constraints. We also summarize the average violation size of each constraint of both algorithms in Figure 3. One can see that with synthetic data, RandRounding actually has a smaller average violation than Iterative. But more importantly, both algorithms are able to find solutions with very small violations compared to the problem size. For example, with as many as 10,000 candidates, the largest violation of a single constraint for both algorithms is only 5.

5.2 Case Study on University Enrollment Data

We provide a case study about student enrollment at UC Berkeley.

Background. Many universities consider fairness (or diversity) as a priority issue in their enrollment process. For example, as stated by UC Berkeley's Strategic Plan for Equity, Inclusion, and

Categories	Attributes	Proportions (%)	α_j (%)	β_j (%)
Gender	Female	50	48	52
	Male	50	48	52
Major	Col. Letters and Science	60	59	60
	Col. Engineering	21	20	21
	Col. Natural Resources	6	5	6
	Col. Chemistry	4	4	5
	Col. Environmental Design	4	4	5
	Sch. Business	5	4	5
Region	Africa	2	2	100
	East Asia and the Pacific	61	50	60
	Europe and Eurasia	13	10	13
	Near East	2	2	100
	South and Central Asia	11	10	11
	Western Hemisphere	11	10	11
Type	Undergraduate	51	50	51
	Graduate	41	40	41
	Transfer	8	7	8

Table 1: Dataset characterization and parameter setting.

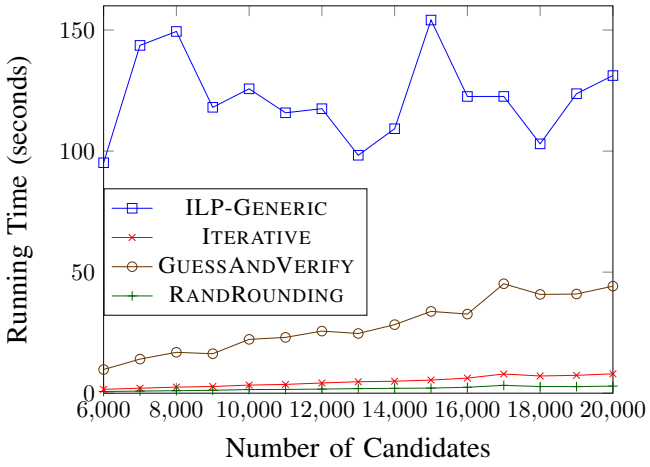

Figure 4: Running time with different problem size.

Diversity, [1] a goal is to "create a critical mass of talented students [...] that will fully represent California's excellence and diversity." At present, UC Berkeley does not represent the diversity of the state. Part of the work of the Division of Equity and Inclusion is to help redress this lack of representation at UC Berkeley.

Data collection. We collect data based on the international student enrollment report (2018) from UC Berkeley. [2] Each student is characterized by four categories: Gender, Major, Region, and Type. In our experiment, for each student, we randomly assign him/her an attribute in each category using the attribute distribution of that category. For proportional fairness constraints, we set each fairness parameter α_j, β_j that is similar to the percentage of this attribute in the corresponding category. We also omit the fairness constraints for attributes that have small percentages in the dataset and relax the lower-bounds in fairness constraints for some attribute that has large proportion in its category. The cardinality threshold in this experiment is simply set as the total number of candidates. See Table 1 for the data statistics and fairness parameter setting.

Experiment results. Figure 4 shows the running times of different algorithms with number of students ranging from 6,000 to 20,000. One can see that all other three algorithms outperform the benchmark ILP-Generic algorithm by a clear margin. Both being exact algorithms, and with 20,000 students, GuessAndVerify is 3-times faster than the generic ILP-Generic algorithm. At the expense of violating some fairness constraints, both RandRounding and Iterative run very fast in practice. From the figure, one can also observe an inconsistency of the running times for ILP-Generic with different problem sizes. The reason to this phenomenon is that the running time of "hard" problem instances, i.e., the one that make ILP-Generic run out of time (we set 300s for the ILP solver as the cut-off time), dominate the average running time of ILP-Generic. For each problem size, ILP-Generic timeouts for at least 29 and at most 46 (out of 100) instances. On the other hand, the other three algorithms can finish every problem instance within the cut-off time of 300s.

[1] https://diversity.berkeley.edu/reports-data/diversity-data-dashboard
[2] https://internationaloffice.berkeley.edu/sites/default/files/student-stats2018.pdf

We also look at the maximum violations of RandRounding and Iterative of each fairness constraint. With 20,000 students, the maximum violations of Iterative and RandRounding are 5 and 3 respectively, both of which are rather small compared to the theoretical bounds we derived in the paper.

6 CONCLUSIONS

In this paper, we study the proportional candidate selection problem in which the fairness constraints are captured by the proportions in terms of the final selection size. We provide both hardness and algorithmic results for this problem. We also conduct experiments over synthetic and real-world data to evaluate the performances of our proposed algorithms.

A natural generalization of this problem considers the weighted case where each candidate has a weight and one wants to maximize the total weights of selected candidates. We remark that both RandRounding and Iterative can be easily modified to address this variation while we leave the design of efficient exact algorithms for the weighted case as an future work. Another interesting future direction is to consider the incentives of the candidates to reveal their true attributes in order to increase their chances to be selected. In addition, a standard scenario of our problem setting is the enrollment for a single school. In future research, it would be interesting to look at assignment problems with proportional fairness constraints for multiple schools.

Acknowledgements

This work is partially supported by the Big Data Intelligence Centre and the Deep Learning Research and Application Centre in The Hang Seng University of Hong Kong.

REFERENCES

[1] Atila Abdulkadiroğlu. 2005. College admissions with affirmative action. *International Journal of Game Theory* 33, 4 (2005), 535–549.
[2] Itai Ashlagi, Amin Saberi, and Ali Shameli. 2019. Assignment Mechanisms under Distributional Constraints. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 229–240.
[3] Haris Aziz. 2019. A Rule for Committee Selection with Soft Diversity Constraints. *Group Decision and Negotiation* (2019).

[4] Haris Aziz, Serge Gaspers, Zhaohong Sun, and Toby Walsh. 2019. From Matching with Diversity Constraints to Matching with Regional Quotas. In *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 377–385.

[5] Nawal Benabbou, Mithun Chakraborty, Xuan-Vinh Ho, Jakub Sliwinski, and Yair Zick. 2018. Diversity Constraints in Public Housing Allocation. In *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 973–981.

[6] Péter Biró, Tamás Fleiner, Robert W. Irving, and David F. Manlove. 2010. The College Admissions problem with lower and common quotas. *Theoretical Computer Science* 411, 34 (2010), 3136 – 3153.

[7] Robert Bredereck, Piotr Faliszewski, Ayumi Igarashi, Martin Lackner, and Piotr Skowron. 2018. Multiwinner Elections With Diversity Constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. 933 – 940.

[8] L. Elisa Celis, Lingxiao Huang, and Nisheeth K. Vishnoi. 2018. Multiwinner Voting with Fairness Constraints. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. 144–151.

[9] Cowen Institue. 2011. Case Studies of School Choice and Open Enrollment in Four Cities.

[10] Lars Ehlers, Isa E. Hafalir, M. Bumin Yenmez, and Muhammed A. Yildirim. 2014. School choice with controlled choice constraints: Hard bounds versus soft bounds. *Journal of Economic Theory* 153, C (2014), 648–683.

[11] Piotr Faliszewski, Piotr Skowron, Arkadii Slinko, and Nimrod Talmon. 2017. Multiwinner voting: A new challenge for social choice theory. In *Trends in Computational Social Choice*, Ulle Endriss (Ed.). AI Access Foundation, Chapter 2.

[12] Piotr Faliszewski, Arkadii Slinko, and Nimrod Talmon. 2017. The Complexity of Multiwinner Voting Rules with Variable Number of Winners. *CoRR* abs/1711.06641 (2017).

[13] Daniel Fragiadakis and Peter Troyan. 2017. Improving matching under hard distributional constraints. *Theoretical Economics* 12, 2 (2017), 863–908.

[14] Isa E. Hafalir, M. Bumin Yenmez, and Muhammed A. Yildirim. 2013. Effective affirmative action in school choice. *Theoretical Economics* 8, 2 (2013), 325–363.

[15] Richard M. Karp. 1972. Reducibility among Combinatorial Problems. In *Proceedings of a symposium on the Complexity of Computer Computations*. 85–103.

[16] D. Marc Kilgour. 2016. Approval elections with a variable number of winners. *Theory and Decision* 81, 2 (2016), 199–211.

[17] Tamás Király, Lap Chi Lau, and Mohit Singh. 2012. Degree bounded matroids and submodular flows. *Combinatorica* 32, 6 (2012), 703–720.

[18] Fuhito Kojima, Akihisa Tamura, and Makoto Yokoo. 2018. Designing matching mechanisms under constraints: An approach from discrete convex analysis. *Journal of Economic Theory* 176 (2018), 803 – 833.

[19] Ryoji Kurata, Naoto Hamada, Atsushi Iwasaki, and Yokoo Makoto. 2017. Controlled School Choice with Soft Bounds and Overlapping Types. *J. Artif. Intell. Res.* 58 (2017), 153–184.

[20] Jérôme Lang and Piotr Skowron. 2018. Multi-attribute proportional representation. *Artificial Intelligence* 263 (2018), 74 – 106.

[21] Lap Chi Lau, R. Ravi, and Mohit Singh. 2011. *Iterative Methods in Combinatorial Optimization* (1st ed.). Cambridge University Press.

[22] Thành Nguyen and Rakesh Vohra. 2017. Stable Matching with Proportionality Constraints. In *Proceedings of the ACM Conference on Economics and Computation (EC)*. 675–676.

[23] Takamasa Suzuki, Akihisa Tamura, and Makoto Yokoo. 2018. Efficient Allocation Mechanism with Endowments and Distributional Constraints. In *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 50–58.

[24] Yongjie Yang and Jianxin Wang. 2018. Multiwinner Voting with Restricted Admissible Sets: Complexity and Strategyproofness. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. 576–582.