Mathematical Programming formulations for the efficient solution of the k-sum approval voting problem

Diego Ponce[1], Justo Puerto[1], Federica Ricca[2], Andrea Scozzari[3]

[1]IMUS and Dep. Estadística e Investigación Operativa,
Universidad de Sevilla
dponce@us.es, puerto@us.es
[2]Universitá di Roma, La Sapienza
federica.ricca@uniroma1.it
[3]Universitá degli Studi Niccoló Cusano, Roma
andrea.scozzari@unicusano.it

September 21, 2018

Abstract

In this paper we address the problem of electing a committee among a set of m candidates and on the basis of the preferences of a set of n voters. We consider the approval voting method in which each voter can approve as many candidates as she/he likes by expressing a preference profile (boolean m-vector). In order to elect a committee, a voting rule must be established to 'transform' the n voters' profiles into a winning committee. The problem is widely studied in voting theory; for a variety of voting rules the problem was shown to be computationally difficult and approximation algorithms and heuristic techniques were proposed in the literature. In this paper we follow an Ordered Weighted Averaging approach and study the k-sum approval voting (optimization) problem in the general case $1 \leq k < n$. For this problem we provide different mathematical programming formulations that allow us to solve it in an exact solution framework. We provide computational results showing that our approach is efficient for medium-size test problems (n up to 200, m up to 60) since in all tested cases it was able to find the exact optimal solution in very short computational times.

Keywords: approval voting, Ordered Weighted Averaging (OWA), k-sum optimization problems.

1 Introduction

A typical problem in collective decision making is to select one (or more) winners among a set of m candidates on the basis of the vote of a set of n voters. This situation arises in many different real-life contexts, as in sport competitions to select the set of winners, or in political elections, and, more in general, whenever a committee must be formed from a larger set of candidates to represent the voters (for example, in companies or universities). For $m > 1$ the problem is a *multi-winner* one and the ballot gives the possibility to each voter to express her/his preference for each single candidate by *approving* or not her/his nomination. This means that voters can approve as many candidates as they like by their preference profile (*approval balloting*). Approval voting is a well-known method used for this kind of multi-winner elections.

The method, introduced by Brams and Fishburn in 1978 [4], was widely studied in the literature on voting theory (see [5, 6, 14] and the references therein).

Consider a set N of n voters and a set A of m candidates. For each i, $P_i = (p_{i1}, \ldots, p_{ij}, \ldots, p_{im})$ denotes the preference profile of voter i which corresponds to a boolean m-vector whose generic element p_{ij} is equal to 1 if candidate j is approved by i and equal to 0 otherwise. Therefore in the problem input we have also a set P of preference profiles of the voters, so that a generic instance can be denoted by (N, A, P). The problem is to find in A the "best" subset of candidates (*winning* or *elected* committee), according to a certain voting rule (criterion).

When we have a single-winner election, the most accepted voting rule is to elect the candidate that has been the most approved, i.e., the one which received the largest number of votes (with a tie-breaking mechanism if needed).

For multi-winner elections, several voting rules have been proposed for approval voting [13]. For a majority of the voting rules computing a winning committee is a difficult problem [2, 10]. Among the many, there is a class of rules known as *centralization procedures* that was widely studied in the literature. Two rules in this class were mainly analyzed, namely, one based on the *minisum* criterion and one on the *minimax* criterion. According to the first criterion, the winning committee corresponds to the subset of candidates in A that minimizes the sum of the n Hamming distances to the preferences profiles of the n voters. The second criterion selects the subset of candidates that minimizes the maximum of its Hamming distances to the voters' profiles. Recently, a new family of rules has been proposed to generalize minisum and minimax in [1], where the authors introduce a family of voting rules which makes use of *Ordered Weighted Averaging* operators (OWA) [21]. In this setting, a vector of n weights $W = (w_1, \ldots, w_n)$ is fixed; then the n distances between voters' profiles and the decision vector (committee) are ordered from largest to smallest and they are weighted with the corresponding weight in W. Clearly, when $W = (1, 0, \ldots, 0)$ we have the minimax criterion, while for $W = (\frac{1}{n}, \ldots, \frac{1}{n})$ we have the minisum criterion. Many other criteria can be defined in this way by tuning the weights in W according to the specific goal of the application. An interesting class of problems arises when vector W has only 0/1 values and more than one weight equal to 1. Suppose to have k elements equal to 1; when they are in the first k positions of the vector of weights they refer to the k largest distances, thus providing what is known in the literature as the k-sum approach already applied to many other combinatorial problems [18]. We have a similar problem when we have elements equal to 1 in the last h positions of the weighting vector (h smallest distances). Both problems have meaningful applications in approval voting.

In this paper we study the problem of selecting a committee by applying approval voting and basing on a k-sum objective function (k-sum approval voting problems). In [1] it is proved that for $1 \leq k < n$ the problem is NP-hard and, therefore, an approximation algorithm is provided to get feasible solutions with guaranteed bounds. On the other hand, the same authors provide polynomial time exact algorithms for some families of weighting vectors that consider the h smallest distances (h fixed). In the present paper we study these kind of problems under a mathematical programming viewpoint, providing different exact formulations for the k-sum approval voting problem, with $1 \leq k < n$. We then exploit these formulations to develop exact solution procedures that may be used to solve medium-size problems at optimality, or to efficiently find a sub-optimal solution when the size of the problem is too large. To develop such formulations we rely on the general approach for solving k-sum optimization problems provided in [19, 20] and on results in [3]. We experimentally study the solution of our k-sum approval voting problem by using the above formulations in a Branch & Bound framework. All formulations were tested on a variety of medium-size randomly generated test problems, in all cases providing the exact optimum in very short times. In view of this, our formulations also

provide an efficient tool to certify optimality of a solution.

We apply the mathematical programming approach also when the h smallest distances are considered in the objective function. We formulate this problem as a polynomial sequence of linear program, thus providing a formal proof that it can be solved in polynomial time as already established in [1].

The paper is organized as follows. Section 2 formally defines the problem and sets the notation. Section 3 presents our mathematical programming formulations for the k-sum approval voting problem. We have developed two different types of formulations. The first ones based on an exponential number of constraints that can be separated efficiently (see Section 3.1) and the second ones based on compact representations of k-sums (see Section 3.2). In Section 4 we describe how all the above mentioned formulations can be strengthened with variable fixing and valid inequalities. The computational experiments are reported in Section 5. There we compare the performance of the formulations on two different set of instances showing its usefulness in solving the problem for instances of medium to large sizes. Finally, Section 6 contains our concluding remarks.

2 Problem definition and basic results

Consider an instance of the k-sum approval voting problem (N, A, P). For a given committee x (i.e., a boolean vector x of length m) we compute the Hamming distance between x and each voter profile P_i, $i = 1, \ldots, n$ thus obtaining the Hamming distance $d_i(x)$ for each voter i. Following the OWA approach, we consider a family of functions, parameterized by a vector of length n that maps a vector of distances $(d_1(x), \ldots, d_n(x))$ to an aggregated function $D(x)$ (OWA score). The k-sum approval voting problem can be stated as follows: select a committee x minimizing the OWA score of Hamming distances $D(x)$. In this paper we study two families of k-sum approval voting problems. The first computes the OWA score using the following vector of weights:

$$W(k) = (1, \ldots, 1, 0, \ldots, 0),$$

where k refers to the number of ones in the first k positions of vector W (electing a committee that minimizes the sum of the k largest Hamming distances).
The second family uses the following weighting vector:

$$M(n-h) = (0, \ldots, 0, 1, \ldots, 1),$$

where h weights equal to 1 are in the last h positions (electing a committee that minimizes the sum of the h smallest Hamming distances). Clearly, the cases $k = h = n$ are the same, and, in this case, we have the same OWA problem, that, in fact, corresponds to the minisum problem.

In [7] (see Proposition 4) the authors account for why the minisum problem is polynomially solvable, but they do not provide a formal proof. The key idea is that the minisum winning committee, in particular under a cardinality constraint on the size of the committee fixed to C, corresponds to a set of C candidates receiving the most votes. In the following, using Linear Programming (LP), we give a formal proof that the problem is polynomial when $k = n$. Denote by γ_j the number of votes for candidate j, $j = 1, \ldots, m$, we have

$$\gamma_j = \sum_{i=1}^{n} p_{ij}.$$

Consider the case when the size of the committee is not given. A valid formulation can be obtained basing on the following observation (see [7]): when $k = n$, all voters' Hamming distances $(d_1(x), \ldots, d_n(x))$ are considered in the objective function, so that a candidate j is elected if the number of votes for her/him γ_j is greater than $n - \gamma_j$. This leads to the following model

$$\begin{aligned} \min \ & \sum_{j=1}^{m}(n-\gamma_j)x_j + \sum_{j=1}^{m}\gamma_j(1-x_j) \\ \text{s.t.} \ & x \in \{0,1\}^m. \end{aligned}$$

Since the objective function is linear, the optimal solution is attained at some vertex of the m-dimensional hypercube. Then, we can relax the binary variables of the above problem obtaining the LP model

$$\begin{aligned} \min \ & \sum_{j=1}^{m}(n-\gamma_j)x_j + \sum_{j=1}^{m}\gamma_j(1-x_j) \\ \text{s.t.} \ & 0 \le x \le 1. \end{aligned}$$

As in [7], we can also consider a constraint on the size C of the committee, obtaining the following model

$$\begin{aligned} \min \quad & \sum_{j=1}^{m}(n-\gamma_j)x_j + \sum_{j=1}^{m}\gamma_j(1-x_j) \\ \text{s.t.} \quad & \sum_{j=1}^{m} x_j = C \\ & x \in \{0,1\}^m. \end{aligned} \tag{1}$$

Since the constraint matrix is again Totally Unimodular (TU), the problem can be still solved by Linear Programming techniques, after relaxing the binary constraints on the variables x (see [17]). This definitely shows that the minisum case is polynomially solvable.

When $k < n$ the k-sum approval voting problem is NP-hard (see [1, 11]). This justifies the idea of studying efficient solution methods for the general problem resorting to heuristic approaches, or approximation algorithms [8, 9, 15, 16]. In the following sections we present a number of exact mathematical programming formulations of the general k-sum approval voting problem with $1 \le k < n$ that can be efficiently solved in a Branch & Bound framework enriched by the use of valid inequalities.

A similar problem arises when we consider the weighting vector $M(n-h)$ with $1 \le h < n$. The computational complexity of this problem was established in [1] when h is not part of the input of the problem. Even if it is shown that the problem is polynomially solvable in this case, to the best of our knowledge, the computational complexity is still open when the problem is formulated with a general h. We discuss this problem in the final section of the paper.

3 Mathematical Programming formulations for the Approval Voting problem

Consider now $W(k)$ with $1 \leq k < n$. Let x be a committee, the Hamming distance, $d_i(x)$, between the profile P_i of voter i and x is given by

$$d_i(x) = \sum_{j=1}^{m} |p_{ij} - x_j|.$$

Since both P_i, $i = 1, \ldots, n$, and x are boolean vectors, we can exploit the fact that the Hamming distance between two boolean vectors corresponds to the vector of the *Exclusive-or* between the vector elements [12]. Thus the Hamming distance $d_i(x)$ can be rewritten as follows

$$\begin{array}{l} d_i(x) = \sum\limits_{j=1}^{m} z_{ij} \\ z_{ij} \geq x_j - p_{ij}x_j \quad j = 1, \ldots, m \\ z_{ij} \geq p_{ij} - p_{ij}x_j \quad j = 1, \ldots, m. \end{array} \tag{2}$$

We can also replace the two inequalities above by the equivalent representation: $z_{ij} \geq x_j(1 - p_{ij}) + p_{ij}(1 - x_j)$. This gives rise to:

$$d_i(x) = \sum_{j=1}^{m} z_{ij} \tag{3}$$

$$z_{ij} \geq x_j(1 - p_{ij}) + p_{ij}(1 - x_j), \quad j = 1, \ldots, m. \tag{4}$$

Let $\sigma_x : \{1, \ldots, n\} \to \{1, \ldots, n\}$ be an ordering function that, for a given x, provides a permutation of the voters' indices such that $d_{\sigma_x(1)}(x) \geq d_{\sigma_x(2)}(x) \geq \ldots \geq d_{\sigma_x(n)}(x)$. For the given permutation the problem of electing a committee that minimizes the sum of the k largest distances can be formulated as follows:

$$\begin{array}{ll} \min & \sum\limits_{h=1}^{k} d_{\sigma_x(h)}(x) \\ & x \in \{0,1\}^m. \end{array} \tag{5}$$

Following the approach in [3], the above problem can be restated as:

$$\min_{x \in \{0,1\}^m} \left(\max \left\{ d_S(x) \mid S \subset \{1, \ldots, n\}, |S| = k \right\} \right), \tag{6}$$

where $d_S(x) = \sum\limits_{i \in S} \sum\limits_{j=1}^{m} |p_{ij} - x_j|$, is the Hamming distance of the set of voters in S to the committee represented by x.

In general k-sum problems, the expression of the contribution of a subset of voters to the election of a candidate can be also computed by means of the γ_j values, namely the number of voters approving a given candidate j. For this purpose, let us introduce some necessary notation. More formally, let S be a set of voters such that $|S| = k$. We define $\gamma_j(S) = \sum_{i \in S} p_{ij}$, as the number of votes of candidate j by the voters in S. For a given x and S such that $|S| = k$, we can compute:

$$d_S(x) = \sum_{j=1}^{m} \max\{\gamma_j(S)(1-x_j), (k-\gamma_j(S))x_j\} \tag{7}$$

$$= \sum_{j=1}^{m} \gamma_j(S)(1-x_j) + \sum_{j=1}^{m} (k-\gamma_j(S))x_j, \tag{8}$$

i.e., the Hamming score of the k voters in S computed w.r.t. a given solution x. Notice that by means of expressions (7) and (8) the score is well calculated even when the solution is not optimal. Basing on these expressions, in the following sections we obtain alternative valid formulations of the k-sum approval voting problem that we then test experimentally in Section 5.

3.1 Valid formulations based on subsets of size k

In this section we propose a first valid formulation for our k-sum approval voting problem which is based on expression (7). We formulate it in the following theorem where, for the sake of simplicity, we avoid specifying $S \subset \{1, \ldots, n\}$ when not necessary.

Theorem 1. *An optimal solution of the k-sum approval voting problem can be obtained solving the following integer programming problem.*

$$\min v \tag{9}$$

$$s.t.\ z_{ij} \geq p_{ij}(1-x_j) \quad \forall i,j \tag{10}$$

$$z_{ij} \geq (1-p_{ij})x_j \quad \forall i,j \tag{11}$$

$$\sum_{j=1}^{m} \sum_{i \in S} z_{ij} \leq v \quad \forall S : |S| = k \tag{12}$$

$$x \in \{0,1\}^m. \tag{13}$$

Proof. Applying (7) in formula (6) gives

$$\min_{x \in \{0,1\}^m} \left\{ \sum_{j=1}^{m} \max\{(k-\gamma_j(S))x_j, \gamma_j(S)(1-x_j)\} | S \subset \{1, \ldots, n\},\ |S| = k \right\}.$$

Defining variables z_{Sj} for all $S \subset \{1, \ldots, n\}$, $|S| = k$, and all j, this is equivalent to

$$\min v \tag{14}$$

$$\text{s.t. } z_{Sj} \geq \gamma_j(S)(1-x_j) \qquad \forall j,\ \forall S : |S| = k \tag{15}$$

$$z_{Sj} \geq (k-\gamma_j(S))x_j \qquad \forall j,\ \forall S : |S| = k \tag{16}$$

$$\sum_{j=1}^{m} z_{Sj} \leq v \qquad \forall S : |S| = k \tag{17}$$

$$x \in \{0,1\}^m. \tag{18}$$

Next, we can define variables z_{ij} for all $i = 1, \ldots, n$ and for all $j = 1, \ldots, m$ and disaggregate each variable $z_{Sj} = \sum_{i \in S} z_{ij}$, which results in

$$\min v \tag{19}$$

$$\text{s.t.} \sum_{i\in S} z_{ij} \geq \gamma_j(S)(1-x_j) \qquad \forall j,\ \forall S : |S| = k \tag{20}$$

$$\sum_{i\in S} z_{ij} \geq (k-\gamma_j(S))x_j \qquad \forall j,\ \forall S : |S| = k \tag{21}$$

$$\sum_{j=1}^{m}\sum_{i\in S} z_{ij} \leq v \qquad \forall S : |S| = k \tag{22}$$

$$x \in \{0,1\}^m. \tag{23}$$

We observe that constraints (20) and (21) for all j and S such that $|S| = k$, can be replaced by the following disaggregated version

$$\begin{aligned}
&\min v \\
&\text{s.t. } z_{ij} \geq p_{ij}(1-x_j) \quad \forall i,j \\
&\quad z_{ij} \geq (1-p_{ij})x_j \quad \forall i,j \\
&\quad \sum_{j=1}^{m}\sum_{i\in S} z_{ij} \leq v \quad \forall S : |S| = k \\
&\quad x \in \{0,1\}^m.
\end{aligned}$$

This concludes the proof. □

Example 1. *We illustrate the above approach reformulating the minimax approval voting problem ($k = 1$) within this general framework.*

$$\begin{aligned}
&\min \quad v \\
&\\
&\quad z_{ij} \geq p_{ij}(1-x_j) \quad \forall i,j \\
&\quad z_{ij} \geq (1-p_{ij})x_j \quad \forall i,j \\
&\quad \sum_{j=1}^{m} z_{ij} \leq v \qquad \forall i \\
&\quad x \in \{0,1\}^m.
\end{aligned} \tag{24}$$

In the following, we develop a second valid formulation for the general k-sum approval voting problem applying (6) but using (8) to represent the Hamming distance instead of (7).

Theorem 2. *The following formulation provides a valid representation of the k-sum approval voting problem.*

$$\min v \tag{25}$$

$$s.t. \sum_{j=1}^{m}(k-\gamma_j(S))x_j + \sum_{j=1}^{m}\gamma_j(S)(1-x_j) \leq v \quad \forall S : |S| = k \tag{26}$$

$$x \in \{0,1\}^m. \tag{27}$$

Proof. Applying in formula (6) the representation (8) instead of (7), the proof follows similarly to that of Theorem 1. □

Since both formulations (9)-(13) and (25)-(27) have an exponential number of constraints, we propose here two different approaches to solve these two models.

A first approach is based on formulation (9)-(13). Let us assume that we embed that formulation in a Branch and Bound scheme and let $(\hat{x}, \hat{z}, \hat{v})$ be the current solution in a node of this Branch & Bound tree.

Procedure for (9)-(13)

- Compute $\hat{r}_i := \sum_{j=1}^{m} \hat{z}_{ij}$ for every i and choose the k largest. Determine S according to the k largest $\hat{r}_i$ for $i \in \{1, \ldots, n\}$.
- Check if $\sum_{i \in S} \hat{r}_i > \hat{v}$. In the affirmative case, we need to add the following constraint related to such S

$$\sum_{i \in S} \sum_{j=1}^{m} z_{ij} \leq v. \tag{28}$$

  Otherwise, i.e. when the answer to the test is no, the current solution is feasible in the current node. Therefore, a valid description of the problem was already available and no more inequalities have to be added.

A similar scheme can be applied to Problem (25)-(27). Let $(\hat{x}, \hat{v})$ be a given feasible solution in a node of its Branch & Bound tree.

Procedure for (25)-(27)

- Compute $\hat{r}_i := \sum_{j=1}^{m} |\hat{x}_j - p_{ij}|$, for all $i = 1, \ldots, n$. Determine S according to the k largest values of $\hat{r}_i$ for $i \in \{1, \ldots, n\}$.
- Check whether $\sum_{i \in S} \hat{r}_i > \hat{v}$. In the affirmative case we need to add the following inequality which is a valid cut that separates $(\hat{x}, \hat{v})$

$$\sum_{j=1}^{m} \gamma_j(S)(1 - x_j) + \sum_{j=1}^{m} (k - \gamma_j(S)) x_j \leq v.$$

Proposition 1. *Formulation (25)-(27) is at least as good as formulation (9)-(13).*

Proof. Let $P_{(25)-(27)}$ and $P_{(9)-(13)}$ be the polyhedra defining the feasible domains of the continuous relaxation of formulations (25)-(27) and (9)-(13), respectively.

Consider a feasible solution (possibly fractional) $(x, v, z) \in P_{(9)-(13)}$. It follows that its projection onto (x, v) belongs to $P_{(25)-(27)}$, and the result follows. □

The above result, together with the fact that formulation (25)-(26) has a smaller number of constraints and variables than formulation (9)-(13), justifies that in our computational experiments (see Section 5) we only report results based on formulation (25)-(26) since its performance is superior to the one of (9)-(13).

3.2 Valid formulations based on Hamming distance among profiles

In this section, we derive alternative valid formulations for the k-sum approval voting problem that do not make explicit use of all the possible subsets of $\{1,\dots,n\}$ of cardinality k. For an arbitrary subset S of $\{1,\dots,n\}$, we consider the sum of the Hamming distances of all P_i, $i \in S$, to x as follows

$$d_S(x) = \sum_{i\in S}\sum_{j=1}^{m} |x_j - p_{ij}|.$$

If there is no confusion, in the following, we simply write d_i in place of $d_i(x)$.

For a given k, with $1 \le k < n$, the problem of electing a committee that minimizes the sum of the k largest Hamming distances can be formulated as a Mixed Integer Linear Programming (MILP) problem, provided that for any given x a permutation σ_x such that $d_{\sigma_x(i)}(x) \ge d_{\sigma_x(i+1)}(x)$, $i = 1,\dots,n-1$, is fixed.

$$\min \sum_{h=1}^{k} d_{\sigma_x(h)}(x) \tag{29}$$

$$z_{ij} \ge x_j(1-p_{ij}) + p_{ij}(1-x_j) \quad i = 1,\dots,n,\ j = 1,\dots,m \tag{30}$$

$$d_i \ge \sum_{j=1}^{m} z_{ij} \quad i = 1,\dots,n \tag{31}$$

$$x_j \in \{0,1\} \quad j = 1,\dots,m. \tag{32}$$

Problem (29)-(32) has m binary variables, nm continuous variables, and $O(nm)$ linear constraints.

This formulation is correct but it is not operational, since it depends on the valid permutation function $\sigma_x(\cdot)$ that sorts the distances in a non-increasing order. However, it is still possible to derive alternative valid formulations that do not make explicit use of that permutation (see [19, 20]). Indeed, let us consider a new variable $t \ge 0$ and a set of n variables v_i, $i = 1,\dots,n$.

Theorem 3. *The following is a valid formulation for the general k-sum approval voting problem.*

$$\min\ kt + \sum_{i=1}^{n} v_i \tag{33}$$

$$s.t.\ v_i \ge d_i - t \qquad i = 1,\dots,n \tag{34}$$

$$d_i \ge \sum_{j=1}^{m} z_{ij} \qquad i = 1,\dots,n \tag{35}$$

$$z_{ij} \ge x_j(1-p_{ij}) + p_{ij}(1-x_j) \qquad i = 1,\dots,n,\ j = 1,\dots,m \tag{36}$$

$$x_j \in \{0,1\} \qquad j = 1,\dots,m \tag{37}$$

$$t \ge 0,\ v_i \ge 0 \qquad i = 1,\dots,n. \tag{38}$$

Proof. Consider the general formulation (6). Following the proof in [20], the inner maximum in problem (6) is equivalent to the following

$$\begin{array}{ll} \max & \sum_{i=1}^{n} d_i q_i \\ & \sum_{i=1}^{n} q_i = k \\ & q_i \in \{0,1\} \quad i = 1,\dots,n. \end{array} \tag{39}$$

The above constraint matrix is TU and thus the integrality constraints on the variables q_i, $i = 1,\dots,n$, in problem (39) can be relaxed to $0 \le q_i \le 1$, $i = 1,\dots,n$, and the resulting problem has the following exact dual:

$$\begin{array}{lll} \min & kt + \sum_{i=1}^{n} v_i & \\ \text{s.t.} & v_i \ge d_i - t & i = 1,\dots,n \\ & v_i \ge 0 & i = 1,\dots,n. \end{array} \tag{40}$$

Notice that, since d_i are distances, the coefficients in the objective function of (39) are non negative and, w.l.o.g., we can set the variable t as $t \ge 0$. To complete the proof, it suffices to add to the above dual problem the constraints (35)-(37). □

When a constraint on the number C of candidates to be elected must be also considered, we can add it to the above program by condition

$$\sum_{j=1}^{m} x_j \le C.$$

An alternative valid formulation for the general k-sum approval voting problem can be provided by exploiting the one proposed in [3], as stated in the following theorem.

Theorem 4. *The following is a valid formulation for the general k-sum approval voting problem.*

$$\min \sum_{i=1}^{n} u_i + \sum_{h=1}^{k} v_h \tag{41}$$

$$s.t.\ u_i + v_h \ge d_i \qquad i = 1,\dots,n,\ h = 1,\dots,k \tag{42}$$

$$d_i \ge \sum_{j=1}^{m} z_{ij} \qquad i = 1,\dots,n \tag{43}$$

$$z_{ij} \ge x_j(1 - p_{ij}) + p_{ij}(1 - x_j) \qquad i = 1,\dots,n,\ j = 1,\dots,m \tag{44}$$

$$x_j \in \{0,1\} \qquad j = 1,\dots,m \tag{45}$$

$$u_i \ge 0 \qquad i = 1,\dots,n \tag{46}$$

$$v_h \ge 0 \qquad h = 1,\dots,k. \tag{47}$$

Proof. Consider the general formulation (6). We introduce the following binary variables y_{ih}, $i = 1,\dots,n$ and $h = 1,\dots,k$, such that, given x, $y_{ih} = 1$ if the distance $d_i(x)$ of voter i is in position $h < k$ in the non-increasing ordering, and $y_{ih} = 0$ otherwise. Following the proof in [3], for a given vector x, the sum of the k largest distances can be written

$$\begin{aligned}
\sum_{h=1}^{k} d_{\sigma_x(h)}(x) = \max \quad & \sum_{i=1}^{n}\sum_{h=1}^{k} d_i y_{ih} \\
\text{s.t.} \quad & \sum_{i=1}^{n} y_{ih} = 1 \quad h = 1,\ldots,k \\
& \sum_{h=1}^{k} y_{ih} = 1 \quad i = 1,\ldots,n \\
& y_{ih} \in \{0,1\} \quad i = 1,\ldots,n,\ h = 1,\ldots,k.
\end{aligned} \tag{48}$$

In fact, problem (48) is an assignment problem, so that we can relax the binary variables $0 \leq y_{ih} \leq 1$. Taking the dual of (48) we obtain

$$\begin{aligned}
\sum_{h=1}^{k} d_{\sigma_x(h)}(x) = \min \quad & \sum_{i=1}^{n} u_i + \sum_{h=1}^{k} v_i \\
\text{s.t.} \quad & u_i + v_h \geq d_i \quad i = 1,\ldots,n,\ h = 1,\ldots,k \\
& u_i,\ v_h \geq 0 \quad i = 1,\ldots,n,\ h = 1,\ldots,k.
\end{aligned} \tag{49}$$

To complete the proof, it suffices to add to the above dual problem the constraints (43)-(45). □

Proposition 2. *The formulations (25)-(27), (33)-(38) and (41)-(47) produce the same LP bound.*

Proof. Let us consider a generic $\hat{x} \in [0,1]^m$. From our discussion above, it is clear that, fixing $\hat{x}$, the objective function value of all problems (25)-(27), (33)-(38) and (41)-(47) equals $\sum_{h=1}^{k} d_{\sigma_{\hat{x}(h)}}(\hat{x})$, where $d_{\sigma_{\hat{x}(1)}}(\hat{x}) \geq \ldots \geq d_{\sigma_{\hat{x}(n)}}(\hat{x})$. Hence, the three problems return the same objective function value for each feasible solution of the continuous polytope and therefore this proves the claim. □

4 Strengthening the formulations

The above MIP formulations are exact but still one can observe that they have some GAP at the root node of the Branch & Bound tree (see Section 5) although this gap is always rather small. The goal of this section is to develop some preprocessing strategies and valid inequalities that allow to improve the polyhedral description of the different formulations, and get a better bound for this GAP with a consequent gain in computational times.

First of all, we advance an easy preprocessing that allows fixing some variables either to zero or to one before the global search starts. The rationale behind that is as follows. For a candidate j to be member of at least one committee it is required that, at least for a subset S of size $k < n$, there is a majority of voters that approves him/her, that is, $\gamma_j(S) \geq \lfloor k/2 \rfloor$. Therefore, if the total number of voters preferring candidate j, γ_j, satisfies $\gamma_j \geq n - \lfloor k/2 \rfloor$ then this candidate will be always included in any committee and thus we can set $x_j = 1$. This justifies (50). On the other hand, if candidate j is only preferred by less than $\lfloor k/2 \rfloor$ voters, she/he will never be

included in a k-sum committee and thus $x_j = 0$. This justifies (51)

$$x_j = 1 \qquad \text{if } \gamma_j \geq n - \lfloor k/2 \rfloor \tag{50}$$

$$x_j = 0 \qquad \text{if } \gamma_j \leq \lfloor k/2 \rfloor. \tag{51}$$

Note that this preprocessing is more interesting for large values of k's.

In the following we also develop a procedure for the efficient solution of the $\hat{k}$-sum approval voting problem for $\hat{k}$ fixed that works in an iterative fashion starting from $k = n$ and solving a k-sum approval voting problem for all $k = n, \ldots, \hat{k}$. This procedure is based on valid inequalities involving optimal objective function values of k-and-$(k+1)$-sum approval voting problems for any k.

First, we analyze whether inequalities (20)-(22) can be adapted to the formulations described in Section 3.2, as valid cuts. Clearly, they are valid inequalities but, as we will see, they do not improve such formulations.

Consider, first, formulation (33)-(38). Note that the cuts in (20), (21) and (22) consist in aggregated forms of constraints (36), (34) and (35), respectively.

$$z_{ij} \geq p_{ij}(1 - x_j) \forall i, j \Rightarrow \sum_{i \in S} z_{ij} \geq \gamma_j(S)(1 - x_j) \quad \forall j$$

$$z_{ij} \geq (1 - p_{ij}) x_j \forall i, j \Rightarrow \sum_{i \in S} z_{ij} \geq (k - \gamma_j(S)) x_j \quad \forall j$$

Hence, the use of them as valid inequalities is not useful. Furthermore $\sum_{j=1}^{m} \sum_{i \in S} z_{ij} \leq kt + \sum_{i \in S} v_i$ can be obtained by means of a natural aggregation of (34) and (35).

In light of the above results, now we develop valid inequalities based on solutions of the k-sum approval voting problem for different k values to be used in a strategy that solves problems for different k consecutively.

In order to present the result some additional notation is required. For a given k, let us denote by $z(k)$ and $x(k)$ the optimal objective function value and an optimal solution of the k-sum approval voting, respectively.

Proposition 3. *For a given instance (N, A, P) of the k-sum approval voting problem the following inequality holds*

$$z(k) \geq \frac{k}{k+1} z(k+1).$$

Proof. By definition, $z(k)$, is the sum of the k largest Hamming distances of the voters' profiles with respect to $x(k)$. It means that distance in position $k+1$, $d_{(k+1)}(x(k))$, satisfies

$$0 \leq d_{(k+1)}(x(k)) \leq \frac{z(k)}{k}.$$

Thus, we can conclude that $x(k)$ is a feasible solution for the $(k+1)$-sum problem and moreover, there exists an upper bound for $z(k+1)$ given by

$$z(k+1) \leq z(k) + \frac{z(k)}{k}. \tag{52}$$

The above expression gives a lower bound for $z(k)$, provided that $z(k+1)$ is known

$$z(k) \geq \frac{k}{k+1} z(k+1). \tag{53}$$

□

Since, for a given (N, A, P) we can solve the different k-sum voting problems in any order, after the above result, it is advisable to do it following the non-increasing sequence $k = n, \ldots, 1$. Indeed, as shown in Section 2, solving the problem for $k = n$ (i.e., the minisum problem) is polynomial. Once this solution and objective function value are found, they can be used to improve the solution for $k = n - 1$ and so on.

The following is an iterative scheme for efficiently solving the k-sum approval voting problem following the strategies illustrated above in this section. This approach has been effectively used in our computational experiments.

1. Solve the problem for $k = n$, i.e. the minisum problem. Its optimal solution, $x(n)$, is easily seen to be

$$x_j = 1 \qquad \text{if } \gamma_j \geq n - \lfloor n/2 \rfloor \tag{54}$$
$$x_j = 0 \qquad \text{if } \gamma_j < \lfloor n/2 \rfloor \tag{55}$$

   Next, obtain $z(n)$, the optimal objective function value of this problem.

2. From $k = n - 1$ to $k = 1$ set the following valid inequalities:

$$\frac{k}{k+1} z(k+1) \leq z(k) \leq z(k+1) - d_{(k+1)}(x(k)).$$

From the discussion above, it is clear that after solving the problem with $W(k+1)$ we have the lower bound on $z(k)$ (53) that we can use as a valid inequality when solving the problem with weighting vector $W(k)$. On the other hand, (52) provides an upper bound on $z(k)$ by $z(k+1)$. We will show in our computational experiments section that the improvements obtained by the application of these strategies are remarkable (see Section 5).

5 Computational results

This section reports on the results of an exhaustive computational test carried out on two sets of instances and our three formulations with and without improvements. We have tested data sets generated according to the scheme proposed in [15]. That paper distinguishes between *uniform* and *biased* data. The former refers to equal probability distribution for 0 and 1 in the profiles, whereas the latter indicates different probabilities for them. Overall, we have solved 22750 instances with the different combinations for n, m, k and uniform and biased data.

In a preliminary analysis, we wanted to test the performance of our three formulations in instances of moderate size ($n = 50$ and different values for $m = 30, 35, 40, 45, 50, 55, 60$), in order to make the decision of which are the formulations to be tested with larger instance sizes. Tables 1 and 2 report the average results of all the 1750 instances for $n = 50$ for the uniform and biased data (5 randomly generated instances for each m and $k = 1, ..., 50$). The results are organized as follows. Each row reports information for a different formulation that is indicated

by its references. For each formulation we include information about average and maximum time (Time(s)), average and maximum percent gap at the root node (GAP (%)), average and maximum number of nodes in the searching tree (Nodes), percentage of instances solved at the root node (%Solved root) and percentage of binary variables fixed with the preprocessing (% Fixed).

	Time(s)		GAP (%)		Nodes		%Solved	%Fixed
Form.	Avg	Max	Avg	Max	Avg	Max	root	
(25)-(27)	65.63	4396.86	0.22	2.63	7388.30	1092091.00	15.89	13.92
(33)-(38)	0.13	2.24	0.22	2.63	276.67	23170	62.74	13.92
(41)-(47)	0.58	19.99	0.22	2.63	1128.03	158152.00	54.17	13.92

Table 1: Summary for $n = 50$ for uniform data

	Time(s)		GAP (%)		Nodes		%Solved	%Fixed
Form.	Avg	Max	Avg	Max	Avg	Max	root	
(25)-(27)	12.34	258.85	0.35	3.12	973.61	52028	17.83	29.42
(33)-(38)	0.06	0.40	0.35	3.13	15.76	959	68.91	29.42
(41)-(47)	0.17	1.23	0.35	3.13	71.33	10914	54.86	29.42

Table 2: Summary for $n = 50$ for biased data

For the formulation (25)-(27), we also report information on the average and maximum number of cuts (# Cuts), and the average and maximum number of cuts in each node (# Cuts node). This information is relevant to understand the number of constraints, out of the exponentially many in the formulation, which are needed to have a valid representation of the problem in each node of the Branch & Bound tree.

We have also tested empirically, see Table 3, that the gap at the root node coincides for the three formulations. This confirms that the three formulations are equivalent in terms of LP gap and reports about the rather small integrality gaps of these formulations.

The conclusion of the above tables is that formulation (25)-(27) is as stronger as the other two in terms of gap, but its performance is inferior in terms of time and number of problems solved at the root node. For this reason, we have decided to carry out the final test for larger instances only with formulations (33)-(38) and (41)-(47).

Next, we compare the performance of formulations (33)-(38) and (41)-(47) for the instances with $n = 100$. Tables 4 and 5 report our results for the two types of data sets i.e., uniform and biased. All the information is organized as in previous Tables 1 and 2.

Data	Avg # Cuts	Max # Cuts	Avg # Cuts node	Max # Cuts node
Uniform	97963.15	9428791	124.43	463
Biased	7390.41	315094	63.80	444

Table 3: Valid inequalities generated in a Branch & Bound tree for solving formulation (25)-(27) with $n = 50$ and for uniform and biased data

	Form.	Time(s) Avg	Time(s) Max	GAP (%) Avg	GAP (%) Max	Nodes Avg	Nodes Max	%Solved root	% Fixed
m=30	(41)-(47)	3.35	23.66	0.17	0.84	1565.53	28751	46.0	9.32
	(33)-(38)	0.37	4.04	0.17	0.84	1488.07	36793	46.4	9.32
m=35	(41)-(47)	4.39	55.32	0.17	2.50	2695.97	126951	43.8	8.87
	(33)-(38)	0.48	10.47	0.17	2.50	2334.13	96381	44.8	8.87
m=40	(41)-(47)	5.40	93.11	0.14	1.14	4149.20	158828	44.4	8.42
	(33)-(38)	0.70	12.53	0.14	1.14	3674.94	121300	45.0	8.42
m=45	(41)-(47)	13.40	251.75	0.13	2.00	11463.60	657000	42.6	9.44
	(33)-(38)	1.56	65.41	0.13	2.00	9924.58	578547	43.6	9.44
m=50	(41)-(47)	18.30	532.31	0.12	0.93	19495.08	781787	43.6	12.38
	(33)-(38)	2.48	78.21	0.12	0.93	17922.55	605694	45.2	12.38
m=55	(41)-(47)	57.27	1652.28	0.10	0.49	53060.66	1875622	41.0	7.55
	(33)-(38)	5.12	131.90	0.10	0.49	45278.16	1256747	40.2	7.55
m=60	(41)-(47)	41.62	2306.34	0.11	1.56	68105.88	7640315	40.0	10.30
	(33)-(38)	5.78	382.24	0.11	1.56	47840.94	4036987	39.6	10.30

Table 4: Summary for n=100 for uniform data

The conclusions from Tables 4 and 5 are the following. Formulation (33)-(38) is one order of magnitude faster than (41)-(47). For instance, the average time for the largest instance sizes ($n = 100, m = 60$), solved with (33)-(38), is of 5.78 seconds and the maximum cpu time was 382.24 seconds. The same instances solved with (41)-(47) take an average time of 41.62 and a maximum of 2306.34 seconds. This fact can be explained by the smaller number of variables and constraints that are needed in formulation (33)-(38) with respect to (41)-(47). The remaining factors (GAP, Nodes, %Solved and % Fixed) are quite similar in both cases. In fact, as we have seen theoretically, both formulations are equivalent in terms of LP gap and they always fix the same number of binary variables. It is also very interesting to test the practical performance of a naïve approximation algorithm based on using the solutions of the linear relaxation, as proposed for instance in Amanatidis et al [1]. One can easily bound from above the empirical performance ratio, $\frac{LPval}{optval}$, of any of our formulations based on the relative gap ($Rgap := \dfrac{optval - LPval}{optval} * 100\%$). Indeed, $\frac{optval}{LPval} \leq \frac{100}{100-Rgap}$. Actually, since the largest relative gap is below 3.13% (see Table 4), this results, in the worst case ($n = 100$, $m = 30$), with an empirical performance ratio bounded above by 1.03.

Next, we have tested our best formulations, namely (33)-(38), in order to explore the size limit that can be solved within 7200 seconds. Tables 6 and 7 report our results. As can be observed in these tables, there is a difference in the performance with respect to uniform and biased data. For the uniform data we get to the time limit already for $n = 150$, whereas for the biased data we are able to solve to optimality all instances for $n = 200$. The reason for this clear difference rests on the fact that the preprocessing, (50) and (51), is much more efficient for biased data where many variables are fixed either to zero or to one. Indeed this percentage is on average of 24.79% for biased data with $n = 200$ as compared to only 7.34% for the uniform data for $n = 150$.

Finally, Figures 1a and 1b show, for the 35 instances with $n = 100$ (5 randomly generated instances for each value of $m \in \{30, 35, 40, 45, 50, 55, 60\}$), the computing time for solving the

		Time(s)		GAP (%)		Nodes		%Solved	%Fixed
	Form.	Avg	Max	Avg	Max	Avg	Max	root	
m=30	(41)-(47)	0.58	1.96	0.30	3.13	48.83	2215	48.2	28.33
	(33)-(38)	0.08	0.42	0.30	3.13	44.13	2013	48.6	28.33
m=35	(41)-(47)	0.63	2.02	0.28	2.63	77.93	4147	43.6	30.07
	(33)-(38)	0.10	0.93	0.28	2.63	81.37	6788	45.6	30.07
m=40	(41)-(47)	0.76	2.47	0.25	2.38	193.00	9921	42.0	26.62
	(33)-(38)	0.13	0.91	0.25	2.38	177.40	8989	48.4	26.62
m=45	(41)-(47)	0.69	2.23	0.24	2.17	145.96	8026	42.4	27.24
	(33)-(38)	0.12	0.81	0.24	2.17	121.17	5985	46.0	27.24
m=50	(41)-(47)	0.69	2.93	0.23	2.00	263.15	15166	44.6	29.21
	(33)-(38)	0.13	0.93	0.23	2.00	202.81	6125	50.2	29.21
m=55	(41)-(47)	0.81	6.33	0.21	1.23	497.78	25672	46.6	28.69
	(33)-(38)	0.17	3.33	0.21	1.23	443.27	26845	52.2	28.69
m=60	(41)-(47)	1.06	39.10	0.22	1.67	1187.28	221519	35.4	28.51
	(33)-(38)	0.26	24.92	0.22	1.67	1136.49	234681	39.0	28.51

Table 5: Summary for n=100 for biased data

		Time(s)		GAP (%)		Nodes		%Solved	%Fixed
	Form.	Avg	Max	Avg	Max	Avg	Max	root	
m=30	(33)-(38)	0.44	9.12	0.12	1.85	819.95	35683	60.6	29.83
m=35	(33)-(38)	2.80	39.15	0.12	1.59	7962.11	162887	38.5	14.09
m=40	(33)-(38)	6.27	225.75	0.10	1.59	22843.11	1165252	56.1	24.90
m=45	(33)-(38)	21.63	985.73	0.09	1.33	76974.33	4847789	48.6	27.74
m=50	(33)-(38)	42.09	1129.91	0.08	0.93	143305.51	4607534	41.7	24.76
m=55	(33)-(38)	50.93	5070.65	0.09	1.61	186926.77	28259032	40.7	20.79
m=60	(33)-(38)	49.97	5409.05	0.08	0.81	207304.41	29489376	56.8	31.45

Table 6: Summary for n=200 for biased data

problem (Figure 1a) and the number of instances solved at the root node, without branching (Figure 1b), for the different $k = 1, \ldots, 100$, and uniform data. We have observed that the behavior is similar for the different values of n. For that reason, we have only included those corresponding to $n = 100$.

Analogously, Figures 1c and 1d show for the biased data instances with $n = 100$ and for the different $k = 1, \ldots, 100$, the computing time for solving the problem (Figure 1c) and the number of instances solved at the root node, without branching (Figure 1d).

Analyzing the figures we conclude that the behavior is rather similar for uniform and biased data. Regarding computing time for solving the problems we observe that it increases when k decreases from $k = n$ until a certain threshold (which depends of the type of data, namely $k \in (15, 30)$ for uniform and $k \in (9, 20)$ for biased) and then the time decreases with k until $k = 1$. This general trend can be explained from the combinatorics of the objective function which relates to $\binom{n}{k}$. With respect to the number of instances solved at the root node, the performance is also similar. This number decreases with k from $k = n$ until a certain value

	Form.	Time(s) Avg	Time(s) Max	GAP (%) Avg	GAP (%) Max	Nodes Avg	Nodes Max	%Solved root	%Fixed
m=30	(33)-(38)	1.20	11.01	0.15	0.93	4302.39	60713	28.27	7.44
m=35	(33)-(38)	2.74	45.11	0.15	2.38	11338.78	237964	30.80	6.97
m=40	(33)-(38)	7.87	138.90	0.13	0.62	36490.51	768328	28.80	7.43
m=45	(33)-(38)	12.64	302.69	0.12	1.92	56540.63	1886774	27.33	7.61
m=50	(33)-(38)	44.67	777.99	0.11	0.89	211780.07	4787806	28.67	7.43
m=55	(33)-(38)	81.99	4068.90	0.10	1.08	419709.27	27576236	23.87	7.48
m=60	(33)-(38)	209.87	6664.97	0.10	0.76	1023055.10	36286662	24.67	6.99

Table 7: Summary for n=150 for uniform data

(which again depends on the type of data, $k \approx 70$ and $k \approx 28$, for uniform and biased data, respectively) and then it increases with k up to $k = 1$.

6 Concluding Remarks

To conclude the paper, we resort to the problem of minimizing the sum of the h smallest Hamming distances already introduced in Section 2. In [1] this problem has been already considered in the approval voting application context, and the authors prove that, when the OWA vector is non-decreasing, that is, the weighting vector is of the form $M(n-h) = (0, \ldots, 0, 1, \ldots, 1)$, with h the number of ones, $1 \leq h < n$, the winning committee can be found in polynomial time for a fixed value of h. They suggest an enumerative approach based on the solution of $\binom{n}{n-h} = \binom{n}{h}$ minisum problems that is obviously not efficient even for a fixed h, and not polynomial if h is part of the input. As already done in Section 2 for the minisum problem, here we can provide a proof based on Linear Programming that formally justifies the polynomial time approach in [1].

For a fixed h, the general problem can be stated as

$$\min_{x \in \{0,1\}^m} \big(\min \big\{ d_S(x) \mid S \subset \{1, \ldots, n\}, |S| = h \big\} \big). \tag{56}$$

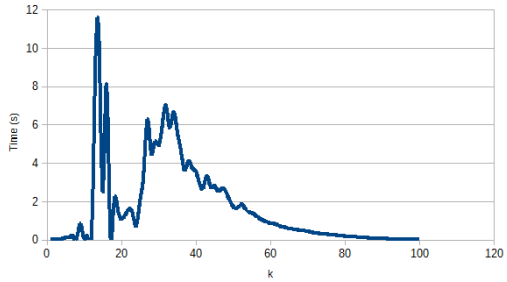
We can switch the two *min* operators, thus obtaining

$$\min_{S \subset \{1,\ldots,n\}, |S|=h} \big(\min_{x \in \{0,1\}^m} \big\{ d_S(x) \big\} \big). \tag{57}$$

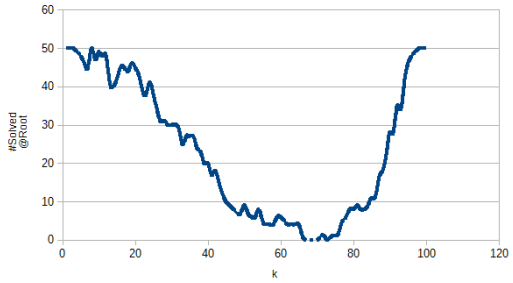
For a given subset $S \subset \{1, \ldots, n\}$, the inner minimum in problem (57) corresponds to the minisum problem

$$\begin{aligned} \min \ & \sum_{j=1}^{m} \gamma_j(S)(1 - x_j) + \sum_{j=1}^{m} (h - \gamma_j(S)) x_j \\ \text{s.t.} \ & 0 \leq x \leq 1, \end{aligned}$$

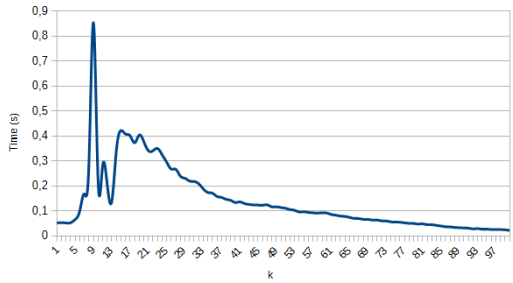
which is polynomially solvable. Then, considering all the $\binom{n}{n-h} = \binom{n}{h}$ subsets of cardinality $n - h$, for a fixed h, the problem can be solved by a sequence of LPs.



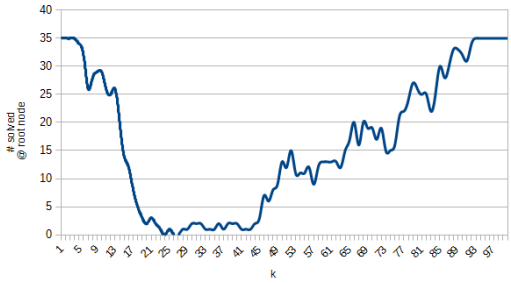
(a) Time vs. k for uniform data $n = 100$



(b) Solved instances at root node for uniform data $n = 100$



(c) Time vs. k for biased data $n = 100$



(d) Solved instances at root node for biased data $n = 100$

Figure 1: Plots of the efficiency of formulation (33)-(38)

It is worth remarking that the application of this problem to approval voting elections is meaningful. In fact, the problem can be stated as: elect a committee minimizing the sum of the h smallest Hamming distances from the voters profiles. As already observed in [1], the application is significant for small values of $n - h$, say $n - h = 1$ or $n - h = 2$. Actually, in these cases, the assumption is that the first one or two maximum distances do not play a significant role in the selection of the committee, and this is true especially when the population of voters is extremely large. The idea is that there will be always some voters whose preferences are completely disjoint from those of the majority of the others. This is, in fact, a way of considering such voters' profiles as *outliers*. But, in our opinion, there are additional cases in which the application is meaningful, namely, for every choice of h such that $n-h \leq \frac{n}{2}-1$. Under this condition, the approval voting problem consists of taking into account only the preferences of the absolute majority of the voters ($h \geq \frac{n}{2} + 1$), with the aim of selecting the committee corresponding to the boolean vector x^* for which the sum of the Hamming distances w.r.t. the h considered profiles is minimized.

Note that, the two problems following the OWA approaches for approval voting studied in this paper can be seen as two different ways of facing the same problem, but giving more importance to one of the two principles that are at the basis of any democratic election. The problem with weighting vector $W(k) = (1, \ldots, 1, 0, \ldots, 0)$ implements the idea that *representation* must be maximized.

If, on the other hand, one wants to give more importance to *governability*, the minimin approach (with weighting vector $M(n-h) = (0, \ldots, 0, 1, \ldots, 1)$) can be pursued with a suitable choice of h, since it is able to guarantee a strong and cohesive consensus. This strength can be enforced by increasing the value of h. We leave the choice of which is the best voting rule for a

country to its lawmakers, who, according to the specific political and social situation in which the election takes place, will be able to choose the best rule.

Going back to our theoretical considerations, to the best of our knowledge, the computational complexity of the minmin problem with weighting vector $M(n-h) = (0, \ldots, 0, 1, \ldots, 1)$ when h is part of the input is still an open problem. In our opinion this is an interesting issue that will be the focus of our future work.

Acknowledgement

This research has been partially supported by Spanish Ministry of Economy and Competitiveness/FEDER grants numbers MTM2013-46962-C02-01 and MTM2016-74983-C02-01. This paper has been written during a sabbatical leave of the second author in La Sapienza, Universitá di Roma whose support is also acknowledged.

References

[1] G. Amanatidis, N. Barrot, J. Lang, E. Markakis, B. Ries Multiple referenda and multiwinner elections using hamming distances: Complexity and manipulability, *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. International Foundation for Autonomous Agents and Multiagent Systems*, 2015.

[2] H. Aziz, S. Gaspers, J. Gudmundsson, S. Mackenzie, N. Mattei, T.Walsh, Computational aspects of multi-winner approval voting, *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. International Foundation for Autonomous Agents and Multiagent Systems*, 2015.

[3] V. Blanco, S. El Haj Ben Ali, J. Puerto Revisiting several problems and algorithms in continuous location with ℓ_τ norms, *Computational Optimization and Applications*, 58 (2014): 563–595.

[4] S.J. Brams, P. Fishburn, Approval voting, *American Political Science Review*, 72 (1978): 831–847. 1978.

[5] S.J. Brams, P. Fishburn, Going from theory to practice: the mixed success of approval voting, *Social Choice and Welfare*, 25 (2005): 457–474.

[6] S.J. Brams, P. Fishburn, *Approval voting*, Springer, 2007.

[7] S. J. Brams, D. M. Kilgour, M. R. Sanver, A minimax procedure for electing committees, *Public Choice*, 132 (2007): 401–420.

[8] J. Byrka, K. Sornat, PTAS for minimax approval voting, *10th Conference on Web Interactions and Network Economics*, Beijin 2014.

[9] I. Caragiannis, D. Kalaitzis, E. Markakis, Approximation algorithms and mechanism design for minimax approval voting, AAAI 2010.

[10] P. Fishburn, A. Pekec, Approval voting for committees: Threshold approaches, *Technical Report* (2004). 2004.

[11] M. Frances, A. Litman, On covering problems of codes, *Theory of Computing Systems* 30 (1997): 113–119.

[12] S. Givant, P. Halmos, *Introduction to Boolean algebra*, Springer, 2009.

[13] D. M. Kilgour (ch 6 of the handbook), Approval balloting for multi-winner elections, in *Handbook on approval voting*, Laslier and Sanver eds (Chapter 6), Springer, 2010.

[14] J-F. Laslier, M. R. Sanver, Eds. *Handbook on approval voting*, Springer, 2010.

[15] R. LeGrand, E. Markakis, A. Mehta, Some results on approximating the minimax solution in approval voting, *AAMAS 07*, Honolulu 2007.

[16] M. Li, B. Ma, L. Wang, On the closest string and substring problems, *Journal of the ACM*, 49 (2002): 157–171.

[17] G. L. Nemhauser, L. A. Wolsey, *Integer programming and combinatorial optimization*, Wiley, Chichester, 1988.

[18] S. Nickel, J. Puerto *Location theory: a unified approach*, Springer, 2006.

[19] W. Ogryczak, A. Tamir. Minimizing the sum of the k largest functions in linear time. *Inf. Process. Lett.*, 85(3) (2003):117–122.

[20] J. Puerto, A. M. Rodríguez-Chía, A. Tamir Revisiting k-sum optimization problems, *Mathematical Programming Ser. A*, in press.

[21] R. R. Yager On ordered weighted averaging aggregation operators in multicriteria decisionmaking, *IEEE Transactions on systems, Man, and Cybernetics*, 18 (1988): 183–190.