Approaching Rank Aggregation Problems by using Evolution Strategies: the case of the Optimal Bucket Order Problem

Juan A. Aledo*, José A. Gámez†, Alejandro Rosete‡

Abstract

The optimal bucket order problem consists in obtaining a complete consensus ranking (ties are allowed) from a matrix of preferences (possibly obtained from a database of rankings). In this paper, we tackle this problem by using $(1+\lambda)$ evolution strategies. We designed specific mutation operators which are able to modify the inner structure of the buckets, which introduces more diversity into the search process. We also study different initialization methods and strategies for the generation of the population of descendants. The proposed evolution strategies are tested using a benchmark of 52 databases and compared with the current state-of-the-art algorithm LIA_G^{MP2}. We carry out a standard machine learning statistical analysis procedure to identify a subset of outstanding configurations of the proposed evolution strategies. The study shows that the best evolution strategy improves upon the accuracy obtained by the standard greedy method (BPA) by 35%, and that of LIA_G^{MP2} by 12.5%.
Keywords: rank aggregation; preference learning; optimal bucket order problem; evolution estrategies; consensus ranking; metaheuristics; weak order

This is the accepted version of:

Juan A. Aledo, Jose A. Gámez, Alejandro Rosete
Approaching Rank Aggregation Problems by using Evolution Strategies: the case of the Optimal Bucket Order Problem
European Journal of Operational Research, 270(3):982-998 (2018)
https://doi.org/10.1016/j.ejor.2018.04.031

Please, visit the provided url to obtain the published version.

*Department of Mathematics, University of Castilla-La Mancha [juanangel.aledo@uclm.es]
†Department of Computing Systems, University of Castilla-La Mancha [jose.gamez@uclm.es]
‡Universidad Tecnológica de la Habana José Antonio Echeverría (Cujae) [Alejandro.Rosete.Suarez@gmail.com

1 Introduction

Rank aggregation serves as an umbrella term for a series of problems whose goal is to compute a consensus ranking from a dataset containing ordered lists of items or other types of preference information about such items [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Rank aggregation is currently a very active field of research in which many disciplines converge, such as social sciences, mathematics, computer sciences, etc. (see [11, Section 1] and the references therein).

Although the seminal problem in this field is the well-known *Kemeny problem* (KP) [3], there are many variants of the problem and many alternative appellations for them: consensus ranking problem, social choice problem, rank aggregation problem, etc., according to the different types of rankings considered as input/output, the different measures/distances to compute the differences between the rankings and the function to optimize. In addition, several relations between rank aggregation and other operations research problems have been addressed, e.g. the *linear ordering problem* [12, 8] or the *feedback arc set problem* [13, 14].

According to Cook [15], there are two broad classes of approaches (ad hoc and distance-based) for aggregating preference rankings to find a consensus. Here we focus on the *optimal bucket order problem* (OBOP), an optimization problem that belongs to the distance-based approach. In the OBOP [9], it is assumed that several preferences/precedences given by different voters about some items are condensed into a pair order matrix [16] C, where each cell $C(u, v) \in [0, 1]$ may be interpreted as the probability that u precedes v, and it is assumed that $C(u, v) + C(u, v) = 1$ and $C(u, u) = 0.5$. Actually, the input for the OBOP is a pair order matrix, whichever way it was obtained. The aim in the OBOP is to obtain a complete ranking with ties, or equivalently, a *bucket order matrix*, that minimizes the L^1 matrix distance with respect to the input pair order matrix C (see Section 2 for the details).

Among the wide family of rank aggregation problems which produce a bucket order as output, although considering different distances to define the objective function, we can highlight the recent works [17] [18] and [19]. All these proposals work within Kemeny's original axiomatic framework (i.e. the search space of the consensus ranking is formed by all possible rankings with ties, and the distance to be minimized is the Kemeny distance).

The OBOP is a problem of both theoretical and applied interest. Apart from the usual fields of application of rank aggregation problems (e.g. social choice theory, voting, meta-search, consensus decision making, etc.), the OBOP has frequently been applied to seriation problems, e.g. in paleontology [20] and in archaeology [21]), the aggregation of visitors' browsing patterns in web portals [22] and discovering of ordered labels from clickstream data [23], among others.

The OBOP falls into several pattern recognition-based disciplines: *machine learning*, as it tries to learn the consensus of order from a dataset of rankings; *preference learning*, as it can also accommodate input data coming from collections of pairwise preferences (e.g. statistics computed from rating data); and *optimization*, as it is defined as the search for the optimal bucket order given

the data.

In particular, in this paper we focus on the optimization aspect of the problem. Given that the OBOP is NP-hard [9], the standard techniques to approach the OBOP in the literature are greedy heuristics. More specifically, the use of the bucket pivot algorithm (BPA) [9, 16] can be considered as the standard option. In fact, some recently-presented improved BPA methods can now be considered as state-of-the-art [11].

In this paper we propose to use metaheuristics to tackle the OBOP. In particular, we choose evolution strategies (ES) [24] as the method to guide the search. We consider that ES are especially appropriate for the OBOP, as we can use greedy solutions to seed the search process and mutation operators to explore their neighborhoods. Our contributions in this paper are:

- we design several mutation operators at the level of items and buckets;
- we study three different initialization methods;
- we propose three different strategies to select the mutation operator to be applied;
- we test the designed the different configurations for the designed ES algorithm using a benchmark composed of 46 datasets taken from a standard repository (*PrefLib*) and 6 datasets obtained from well-known rating-based preference data (*MovieLens-100k*);
- we carry out a study from three different dimensions: accuracy of the solutions obtained, efficiency of the algorithms designed (time and number of evaluations), and inner structure of the solutions (number of buckets and similarity of the solutions found by the different algorithms);
- the statistical analysis of the experimental results clearly shows a significant difference in favor of the ES with respect to current greedy state-of-the-art algorithms.

The rest of this paper is organized as follows. Section 2 introduces the target problem, namely OBOP, while in Section 3 we describe our metaheuristic-based approach to solve it. Section 4 contains the experimental evaluation carried out using a benchmark of 52 datasets and a thorough analysis of the results in terms of accuracy, efficiency and inner structure of the solutions. Finally, in Section 5 we present our concluding remarks.

2 The optimal bucket order problem

Given a set of items $I = \{1, \ldots, n\}$, a *bucket order* $\mathcal{B}$ [4] is an ordered sequence of k subsets (buckets) $I_1, I_2, \ldots, I_k$ of I, $1 \leq k \leq n$, which is an ordered partition of I (i.e., they are pairwise disjoint and the union of all of them is equal to I). For instance, for $n = 4$ we will write $1|3|2,4$ to denote the bucket order with 3 buckets $I_1 = \{1\}$, $I_2 = \{3\}$ and $I_3 = \{2,4\}$, indicating that 1 precedes the

other three items, 3 precedes 2 and 4, and there is no preference between 2 and 4 (which are *tied*).

In general, by *ranking* we understand any precedence order among (some of) the items in I (possibly with ties). Incomplete rankings do not rank all the items, while in partial rankings there is no precedence relation among some items. According to this, a bucket order is a *partial complete* ranking[1].

Given a bucket order $\mathcal{B}$ and $u, v \in I$, we will write $u \prec_{\mathcal{B}} v$ to express that u precedes v (u is preferred to v). If u and v are tied, we will write $u \sim_{\mathcal{B}} v$. In [16] a bucket order $\mathcal{B}$ is represented by means of a matrix B of dimension $n \times n$:

$$B(u,v) = \begin{cases} 1 & if \quad u \prec_{\mathcal{B}} v \\ 0.5 & if \quad u \sim_{\mathcal{B}} v \quad (\text{ includes } u = v) \\ 0 & if \quad v \prec_{\mathcal{B}} u \end{cases} \tag{1}$$

In particular, $B(u,v) + B(v,u) = 1$ for all $u, v \in I$.

A *pair order matrix* [16] is a matrix C such that $C(u,v) \in [0,1]$ for all $u, v \in I$, with $C(u,v) + C(v,u) = 1$ if $u \neq v$ and $C(u,u) = 0.5$. C may be interpreted as a precedences matrix (possibly obtained from a database of rankings, from a database of ratings, etc.), where $C(u,v)$ expresses the probability of u preceding v. Given two pair order matrices C_1 and C_2, we consider the following L^1 matrix distance:

$$D(C_1, C_2) = \sum_{u \neq v} |C_1(u,v) - C_2(u,v)|. \tag{2}$$

Then, given a pair order matrix C, the *optimal bucket order problem* (OBOP) consists in finding the bucket order $\mathcal{B}$ such that $D(B, C)$ is minimum.

In [25] the distance (2) was characterized as the only distance measure for the set of partial orderings determined by a certain set of axioms. Hence, the OBOP should be reported within this ranking aggregation framework.

2.1 Solving *OBOP* by using greedy algorithms

As the OBOP is an NP-hard problem [9], several greedy heuristic algorithms have been proposed to solve it. Undoubtedly, the bucket pivot algorithm (BPA)[9, 16] can be considered the standard greedy method for dealing with the OBOP.

BPA (Figure 1) receives as input a set of items I and a pair order matrix C. Then, one of the items is chosen at random to be the *pivot* p and, based on a given threshold β, $0 < \beta < 0.5$, the rest of the items are split into three sets: $L = \{u \in I \mid C(p,u) < \frac{1}{2} - \beta\}$, $R = \{u \in I \mid \frac{1}{2} + \beta < C(p,u)\}$ and $S = I \setminus (L \cup R) = \{u \in I \mid \frac{1}{2} - \beta \leq C(p,u) \leq \frac{1}{2} + \beta\}$, which is the central bucket containing the pivot p. The final ranking is generated by recursively calling BPA for the successive L and R buckets.

Although BPA exhibits a good performance when applied to the OBOP [9, 26, 22], it does suffer from certain drawbacks. In fact, the random selection of the initial pivot has a significant impact on the bucket order obtained. Krenke

[1] In the literature, they are also called *weak orders*, *ordered partitions* and *partial linear orders*, among others

BPA(I, C, β)

Input: I, set of items; C, pair order matrix; $\beta \in [0, 0.5]$
Output: Bucket order

```
1   if I = ∅ then return ϵ
2   Pick a pivot p ∈ I at random
3   L ← ∅, S ← {p}, R ← ∅.
4   for all item u ∈ I − {p} do
5       if C(p,u) < 1/2 − β then L ← L ∪ {u}
6       else if 1/2 − β ≤ C(p,u) ≤ 1/2 + β then S ← S ∪ {u}
7       else if 1/2 + β < C(p,u) then R ← R ∪ {u}
8       end if
9   end for
10  return concatenate(BPA(L,C,β), S, BPA(R,C,β))
```

Figure 1: The bucket pivot algorithm (BPA)

et al. [26] propose to reduce this risk by using a two-step approach (BPA-CC): first, the items are clustered into groups by using precedence-based similarity; then, BPA is run over the clusters obtained. To do this, a secondary precedence matrix C', which is defined over the clusters, is computed by collapsing the original one.

More recently, Aledo et al. [11] proposed several BPA-based methods which incorporate a criterion for the selection of promising pivots, and which also use more than one item as pivot when making the decision about where to place the remaining items. In this way, the authors tackle the problem detected in [26] regarding the "danger of ordering entities based solely on their pairwise preferences with respect to the pivot". The best algorithm proposed in [11], namely LIA_G^{MP2}, significantly improves upon both the original BPA and BPA-CC. This algorithm constitutes the current state of the art when solving the OBOP, and it will be used for comparison against our proposals in this study.

3 A metaheuristic approach for the optimal bucket order problem

Given the NP-hardness of rank aggregation problems [27, 28, 29, 9], in addition to heuristic greedy algorithms, several metaheuristics approaches have also been developed to tackle them: genetic algorithms [30, 31], local search (HC, ILS, VNS) [32], ant colony [33] and differential evolution [18]. The successful results obtained by metaheuristic algorithms in other rank aggregation problems led us to apply this type of search engines to the OBOP.

In the OBOP the solution is an ordered partition of $[[n]]$, and the number

of possible partitions is determined by the Fubini number [34],

$$\text{Fub}(n) = \sum_{k=0}^{n} k! S(n,k),$$

where $S(n,k)$ is a Stirling number of second kind, which can be computed recursively via

$$S(n,k) = k \cdot S(n-1,k) + S(n-1,k-1),$$
$$S(n,n) = 1, \;\; S(n,0) = 0.$$

In order to get an idea of the cardinality of the search/solution space of the OBOP, below we show some values of Fub(n):

n	10	50	100	150
Fub(n)	102×10^{6}	1.99×10^{72}	5.56×10^{173}	3.09×10^{286}

In this paper we propose the use of *evolution strategies* (ES) as the metaheuristic search engine to tackle the OBOP. ES are specialized evolutionary algorithms whose most striking quality is their efficiency in terms of time complexity [24]. ES (see for example [35]) are usually only based on mutation as the variation operator, whereas crossover is rarely used. Selection is based on the fitness value. Specifically, in this study we follow the $(1 + \lambda)$-ES, which means that the population has a single parent from which λ offspring/children are generated at each iteration/generation. Then, the λ offspring compete with the parent to replace it for the new iteration, that is, elitist replacement is applied. This scheme has proved to be better than the original $(1, \lambda)$-ES, in which the best of the λ offspring is selected as parent for the next iteration, the current parent always being discarded. Figure 2 shows the pseudocode of the $(1 + \lambda)-$ES (adapted from [35]).

$(1 + \lambda)$ − EvolutionStrategy(I, C, λ)

Input: I, set of items; C, pair order matrix;
λ, the number of offspring
Output: Bucket order
1 Initialize a population with 1 individual (*parent*)
2 Evaluate *parent*
3 **repeat**
4 $\Lambda \leftarrow$ Generate λ offspring from *parent*
5 Evaluate individuals in Λ
6 *parent* $\leftarrow$ best(parent $\cup \Lambda$)
7 **until** the stopping condition holds
8 **return** *parent* // the best solution found

Figure 2: Pseudocode of the $(1 + \lambda)-$ES

The next subsections detail the design of the main components in our proposed $(1 + \lambda)-$ES.

3.1 Individual representation and evaluation

Any potential solution (*bucket order*) $\mathcal{B}$ of the elements in $I = [[n]]$ can be represented as a sorted list of buckets $\mathcal{B} = (I_1, I_2, \ldots, I_k)$, $1 \leq k \leq n$, where each bucket is a non-empty set of items satisfying $I_i \cap I_j = \emptyset$, $i \neq j$ and $\bigcup_{i=1}^{k} I_i = [[n]]$.

For a given bucket order $\mathcal{B}$, the fitness function is given by

$$f(\mathcal{B}) = D(B, C) = \sum_{u \neq v} |B(u, v) - C(u, v)|, \tag{3}$$

where B is the matrix associated with $\mathcal{B}$ according to (1). As stated above, our goal is to find the bucket order $\mathcal{B}$ which minimizes (3).

3.2 Initial solution generation

We selected the following three procedures in order to construct the initial solution, thus considering one random method and two (fast) informed ones.

- BPA: the heuristic bucket pivot algorithm for the OBOP (Figure 1).
- Borda: the Borda count method [36]) generates a permutation as output. However, during the computation some items are tied with respect to the score used to create the ranking, and the method randomly/arbitrarily breaks the ties in order to obtain a permutation. Here we use the Borda count algorithm but without tie-breaking, thus generating a bucket order.
- Rd: a ranking is generated at random. First, we generate a random permutation (one item per bucket). Then, the list is scrolled from the beginning and each item is joined to the previous bucket with a probability of 0.5. Otherwise, the item is included in a new bucket.

Note that BPA (which depends on the *randomly* selected pivot) and Rd may obtain different bucket orders in different calls, while Borda (allowing ties) is a deterministic procedure.

3.3 Mutation operators

First, it should be pointed out that no *repair* operator is used in the proposed evolution strategy. Consequently, all the operators must be *closed*, in the sense that if the input is a bucket order for $[[n]]$, then so is the output.

We propose two types of mutation operators. The first type is a direct adaptation to the level of buckets of the mutation operators defined to deal with permutations. We consider the three most popular permutation-oriented mutations, i.e., insertion, interchange and inversion [35, 12]:

- Bucket insertion: a random bucket is moved from its position to another random one.
  Example: $1, 2|3|4, 5, 6|\mathbf{7}, \mathbf{8} \longrightarrow \mathbf{7}, \mathbf{8}|1, 2|3|4, 5, 6$

- Bucket interchange: two buckets are randomly chosen and their positions are interchanged.
  Example: $\mathbf{1},\mathbf{2}|3|4,5,6|\mathbf{7},\mathbf{8} \longrightarrow \mathbf{7},\mathbf{8}|3|4,5,6|\mathbf{1},\mathbf{2}$

- Bucket inversion: a sublist of consecutive buckets is randomly selected. Then, the order of the buckets in the sublist is reversed.
  Example: $\mathbf{1},\mathbf{2}|\mathbf{3}|\mathbf{4},\mathbf{5},\mathbf{6}|7,8 \longrightarrow \mathbf{4},\mathbf{5},\mathbf{6}|\mathbf{3}|\mathbf{1},\mathbf{2}|7,8$

Note that the above operators do not change the number of buckets, their size or composition; they just change the positions of the buckets in the ranking. In order to obtain greater diversity in the search process, we also introduce four new mutation operators, which modify the inner composition of the buckets:

- Bucket union: two consecutive buckets are randomly selected and joined in a bucket.
  Example: $\mathbf{1},\mathbf{2}|\mathbf{3}|4,5,6|7,8 \longrightarrow \mathbf{1},\mathbf{2},\mathbf{3}|4,5,6|7,8$

- Bucket division: a bucket (with at least two items) is randomly selected and randomly divided into two buckets. Note that there is no order inside a bucket, so the items are randomly placed in any of the two new buckets (probability of 0.5).
  Example: $1,2|3|\mathbf{4},\mathbf{5},\mathbf{6}|7,8 \longrightarrow 1,2|3|\mathbf{4},\mathbf{6}|\mathbf{5}|7,8$

- Item insertion: an item is picked at random from a randomly selected bucket. Then, the item is (randomly) inserted in a different position in the ranking. The item can either be added to an existing bucket or used to create a new (singleton) bucket. The item is never reinserted in the original bucket. If the original bucket had a single element, it would become empty and therefore be removed.
  Example: $1,2|3|\mathbf{4},5,6|7,8 \longrightarrow 1,2,\mathbf{4}|3|5,6|7,8.$

- Item interchange: two different buckets are selected at random. Then, one item from each bucket is randomly selected and the chosen items are interchanged. Example:
  $1,2|\mathbf{3}|\mathbf{4},5,6|7,8 \longrightarrow 1,2|\mathbf{4}|\mathbf{3},5,6|7,8.$

Observe that the neighborhoods associated with the previous operators (especially the ones associated with bucket division, item insertion and item interchange) are rather large. This fact prevents us from using local search algorithms based on a systematic exploration of the neighborhoods; on the contrary, it invites us to choose ES as search engines, as they provide an efficient exploration of the search space.

Finally, it is worth pointing out that there is (always) at least one sequence of the previous mutations that allows the transformation of any given solution into any other.

3.4 Population generation

The last component of our ES is the way in which the mutation operators are selected in order to generate the λ children from the current parent.

We propose to explore three different schemes of mutation operator selection:

- *Random* (*Rd*): for each of the offspring, the mutation to be applied is randomly selected from the set of the (seven) available operators.
- *All* (*All*): the seven mutations are applied in each generation, each one generating at least one offspring. This implies $\lambda \geq 7$.
- *Unique* (*Uni*): all the offspring of each generation are generated with the same mutation operator, which is cyclically changed for each iteration.

Both *Rd* and *All* have a potentially larger neighborhood than *Uni*. Regarding *Uni*, it resembles a variable neighborhood search (see for example [37]).

4 Experimental Study

In this section we describe the datasets used and the experiments carried out in order to evaluate our proposal. All the experiments were ran on standard computers (3 GHz processor and 32 GB RAM) with Linux operating systems. All the algorithms were implemented in Java (sequential implementation) and the processes were allowed to use a maximum of 2 GB of RAM.

4.1 Datasets

We considered a representative benchmark of 52 datasets with sizes (i.e., number n of ranked items) of between 50 and 250. The datasets can be divided into two groups:

- We used the PrefLib[2] repository as our main source. We focused on datasets of type "election data" (ED), in particular the datasets with sizes in the range 50-250. After that, in order to have a manageable number of datasets, we selected a subset according to the following rule: if several datasets have the same number of items (n), then select the one with the higher number of voters (m) (alphanumerical order is used for tie-breaking). After this reproducible filtering process, we obtained 46 datasets.

  Among the different formats available in PrefLib, we work with datasets in pwg format, which basically (after normalization) codify precedence order matrices.

- A usual situation with rank data is that for large values of n the number of voters is small. In fact, the datasets selected from PrefLib have $m = 4$

[2]PrefLib: A library for Preferences [38]. `http://www.preflib.org/`

in most cases, where each voter corresponds to a particular search engine. To deal with datasets with more voters, in [26] the authors propose collecting the statistics from preference data. In this sense, we considered the Movielens[3] 100k dataset, which was developed by GroupLens and contains 100000 ratings $(1, \ldots, 5)$ given by 943 users to 1688 movies. All the users in the dataset rated at least 20 movies.

From the Movielens 100k dataset we generated 6 datasets with $n = 50, 100, 150$, 200 and 250 (two datasets) items (movies). For the sake of reproducibility, below we describe how we carried out the selection of the movies. To create the dataset ML-x-y, we started with the movie numbered y and selected the set of movies $\{y, y+6\cdot 1, y+6\cdot 2, \ldots, y+6\cdot(x-1)\}$. Then we took all the users that rated at least one of these movies and created a dataset of bucket orders: one bucket order by user, where each bucket contains the movies with the same rating $(1, \ldots, 5)$. The obtained bucket orders contain many ties (there are, at most, 5 buckets) and are rather incomplete, since each user only ranks a small percentage of the available movies.

Observe that the number of voters (samples or rankings) has no influence on the experiments' computational complexity, as the dataset information is collapsed into the $n \times n$ precedence order matrix.

Table 1 shows the main features of the datasets considered. In particular, the first three columns are:

- DB: the identifier of the database in the PrefLib repository.
- n: the number of items to sort.
- m: the number of voters/instances/rankings in the dataset.

In addition, the last three columns of Table 1 show the results (fitness) of running the three initialization methods described in Section 3.2. In the case of BPA and Rd, we show the average $\pm$ variance over 30 independent runs. As Borda (allowing ties) is a deterministic algorithm, it has no variance. The best result for each dataset is boldfaced.

According to statistical tests (Friedman followed by a post-hoc test, $\alpha = 0.05$), there is no statistically significant difference between BPA and Borda as initialization methods (p-value 0.43). As expected, these two heuristic algorithms clearly outperform the non-informed initialization method proposed (Rd). The ranking obtained by the Friedman test is Borda (1.42), BPA (1.58) and Rd (3). Borda is better than BPA in 30 out of the 52 datasets and worse in 22 datasets. In Table 1 we can observe how Borda is systematically better than BPA when n is larger (> 100). In other words, as n increases the negative effect of the random pivot selection in BPA is amplified.

[3]Downloaded from `http://grouplens.org/datasets/movielens/100k/`

Table 1: Description of the databases used in the experiments, and the results of applying the three initialization methods.

DB	n	m	Random	Borda	BPA
ED-10-11	50	11	1204.1 ± 52.8	**805.7** ± 0	846.0 ± 60.5
ML-050-6	50	936	1215.9 ± 56.8	695.6 ± 0	**633.3** ± 82.1
ED-10-02	51	9	1254.9 ± 50.2	**778.8** ± 0	822.2 ± 65.0
ED-10-15	52	14	1294.8 ± 67.4	**834.8** ± 0	894.6 ± 56.1
ED-10-03	54	10	1431.0 ± 64.1	**931.4** ± 0	972.6 ± 85.5
ED-15-34	55	4	1484.7 ± 100.7	**631.5** ± 0	669.2 ± 59.6
ED-15-77	56	4	1537.3 ± 109.5	691.3 ± 0	**640.0** ± 42.8
ED-10-16	57	16	1526.8 ± 76.3	955.1 ± 0	**907.9** ± 61.4
ED-15-54	60	4	1741.6 ± 107.2	632.5 ± 0	**611.9** ± 57.9
ED-10-17	61	17	1813.0 ± 94.9	**1122.2** ± 0	1123.9 ± 121.4
ED-10-14	62	15	1854.9 ± 109.6	**1204.8** ± 0	1266.7 ± 104.4
ED-15-11	63	4	1931.4 ± 144.9	712.0 ± 0	**701.7** ± 57.8
ED-15-30	64	4	1971.0 ± 130.0	835.5 ± 0	**813.8** ± 41.1
ED-15-31	67	4	2218.2 ± 170.5	863.5 ± 0	**854.0** ± 72.3
ED-15-35	68	4	2216.0 ± 121.0	987.5 ± 0	**953.1** ± 65.9
ED-15-15	69	4	2340.4 ± 163.3	1060.0 ± 0	**1057.2** ± 91.2
ED-15-16	70	4	2462.3 ± 149.1	**931.0** ± 0	930.1 ± 73.6
ED-15-10	71	4	2481.7 ± 145.0	1098.5 ± 0	**986.8** ± 118.8
ED-15-60	72	4	2548.2 ± 199.2	959.5 ± 0	**920.2** ± 73.6
ED-15-57	73	4	2543.6 ± 139.9	1231.5 ± 0	**1166.0** ± 101.6
ED-15-51	77	4	2902.8 ± 238.6	**1078.0** ± 0	1100.9 ± 76.4
ED-15-69	81	4	3270.6 ± 208.2	**1109.0** ± 0	1128.2 ± 78.1
ED-15-26	82	4	3304.6 ± 140.0	**1248.0** ± 0	1253.0 ± 108.4
ED-15-19	87	4	3746.6 ± 190.0	**1383.5** ± 0	1416.7 ± 68.8
ED-15-39	89	4	3865.5 ± 231.8	**1333.0** ± 0	1376.2 ± 140.1
ED-15-24	91	4	4043.6 ± 251.9	1497.5 ± 0	**1488.1** ± 144.4
ED-15-13	93	4	4214.6 ± 225.1	2120.5 ± 0	**2014.7** ± 258.1
ED-15-27	95	4	4448.6 ± 195.7	1785.5 ± 0	**1755.5** ± 104.4
ED-15-21	96	4	4498.3 ± 271.3	**1787.5** ± 0	1811.1 ± 124.4
ED-15-08	99	4	4765.6 ± 225.2	2057.5 ± 0	**1789.6** ± 165.3
ED-14-02	100	5000	4880.9 ± 144.2	3245.1 ± 0	**3082.8** ± 239.4
ML-100-1	100	941	4851.7 ± 162.7	2808.5 ± 0	**2736.4** ± 262.4
ED-15-28	102	4	5061.9 ± 292.3	2139.6 ± 0	**2017.6** ± 53.2
ED-11-03	103	5	5143.0 ± 328.9	**2130.5** ± 0	2184.9 ± 155.7
ED-15-29	106	4	5554.8 ± 204.0	**2041.0** ± 0	2102.1 ± 165.3
ED-15-07	110	4	5979.6 ± 376.6	2063.0 ± 0	**2060.1** ± 159.0
ED-15-22	112	4	6099.0 ± 357.6	**2448.5** ± 0	2497.5 ± 160.5
ED-15-09	115	4	6544.9 ± 391.4	**2448.0** ± 0	2478.3 ± 156.3
ED-15-20	122	4	7373.5 ± 309.1	**3436.5** ± 0	3440.0 ± 254.9
ED-15-17	127	4	7956.2 ± 346.3	**3246.0** ± 0	3344.2 ± 209.1
ED-15-33	128	4	8071.3 ± 411.7	**3289.5** ± 0	3352.8 ± 244.9
ED-15-40	131	4	8445.2 ± 430.1	**3756.0** ± 0	3777.3 ± 264.1
ED-15-23	142	4	10034.0 ± 403.6	**3973.0** ± 0	3999.2 ± 274.6
ML-150-2	150	931	11017.6 ± 419.8	**6249.6** ± 0	7023.5 ± 519.3
ED-15-32	153	4	11582.2 ± 474.2	**4320.5** ± 0	4417.0 ± 257.5
ED-15-14	163	4	13196.2 ± 638.7	**4887.5** ± 0	5011.9 ± 319.7
ED-10-50	170	4	14182.2 ± 467.2	**6710.2** ± 0	7895.8 ± 474.2
ML-200-3	200	936	19920.9 ± 541.2	**11646.9** ± 0	13273.2 ± 1016.5
ED-11-01	240	5	28538.4 ± 1173.4	7478.8 ± 0	**6399.1** ± 260.8
ED-11-02	242	5	29189.5 ± 905.1	**14818.6** ± 0	14955.8 ± 945.7
ML-250-4	250	942	31018.2 ± 516.7	**22125.5** ± 0	23552.3 ± 1150.6
ML-250-5	250	939	30677.6 ± 578.6	**21770.3** ± 0	23362.3 ± 1223.9

4.2 Experiment design and results analysis

In this section we analyze the performance of our proposal on the selected databases. As we have defined three ways of generating the initial solution (*parent*), and three schemes to generate (by mutation) the population of descendants, we obtain nine configurations of evolution strategies to be tested: {Borda, BPA, Rd} × {Rd, All, Uni}. Furthermore, we include LIA_G^{MP2} in the comparison study. LIA_G^{MP2} is a BPA-based algorithm which uses an informed criterion (LIA) for pivot selection and also considers a multi-pivot strategy to decide where to place the items. This algorithm represents the current state of the art for the OBOP [11][4].

Since the ES are of a stochastic nature, we performed 30 independent runs for each configuration and dataset. As parameters, we set $\lambda = 7$ (to be coherent with the number of mutation operators proposed), and the maximum number of fitness evaluations to $10n^2$.

4.2.1 Accuracy results

In this section we discuss the results obtained by each algorithm regarding *fitness*, i.e., the distance of the obtained bucket orders to the precedence matrix C. These values (averaged over 30 runs) are shown in Table 2, where the best result/s for each dataset is/are highlighted in boldface.

Note that several configurations of the ES achieve the same best fitness solution. Regarding the (averaged) performance, Borda-Uni, Borda-Rd and BPA-Uni obtain the best fitness for the considered benchmark. Nonetheless, to properly evaluate the results, we followed a standard machine learning statistical analysis procedure [39, 40]. In particular, the ExReport tool was used [41].

First, a Friedman test ($\alpha = 0.05$) was performed. The obtained p-value (1.9e-42) rejected the null hypothesis that all the methods are equivalent. Then, the algorithm ranked first by the Friedman test (Borda-Uni) was used as control and a Holm post-hoc test ($\alpha = 0.05$) was applied to discover the outstanding methods. Figure 3.(a) shows the rank distribution for the Friedman test. In particular, the boxes corresponding to algorithms which are not statistically different from Borda-Uni are in white. We observe that the different ES configurations significantly[5] outperform the current state-of-the-art algorithm (LIA_G^{MP2}), which is ranked in 10th position (average position is 9.90). Regarding the ES algorithm, Borda-Uni is significantly better than the rest of the configurations except Borda-Rd.

[4] Remarks:

- LIA_G^{MP2} outperforms Borda and BPA by 21% for the studied benchmark.
- For completeness we also studied the three ES configurations which result from using LIA_G^{MP2} as initial solution for the ES. However, the higher quality of this initial solution does not have a positive impact on the search and so we discarded it as an initial solution (see A for details).

[5] Pairwise post-hoc test confirms this fact.

Table 2: Fitness (distance) results for the 52 datasets. The average ± variance over the 30 runs is shown in the first row, and the best solution out of the 30 runs is shown in the second row.

Dataset	LIA	Rd-Rd	Rd-All	Rd-Uni	Borda-Rd	Borda-All	Borda-Uni	BPA-Rd	BPA-All	BPA-uni
ED-10-11	616.4±11.3	587.1±17.7	591.7±20.5	577.2±9.0	576.1±5.2	573.8±4.8	575.2±1.9	581.4±16.2	573.7±14.4	**572.8**±8.2
	601.7	573.2	569.7	561.0	567.2	**558.0**	572.2	560.4	558.6	**558.0**
ML-050-6	455.6±0.0	419.7±0.7	419.6±0.6	419.2±0.8	419.6±2.3	420.5±4.2	**418.8**±0.8	419.7±1.6	420.9±3.8	419.7±1.6
	455.6	418.6	418.6	**418.2**	**418.2**	**418.2**	**418.2**	**418.2**	**418.2**	**418.2**
ED-10-02	563.3±2.0	548.6±19.7	563.2±30.4	546.6±15.8	542.7±14.5	545.7±13.6	545.5±1.8	**537.3**±10.7	542.0±14.2	539.5±13.4
	556.4	**525.0**	**525.0**	**525.0**	**525.0**	**525.0**	544.4	**525.0**	**525.00**	**525.0**
ED-10-15	606.8±7.6	609.2±38.8	611.4±39.4	583.8±14.8	581.7±7.1	585.3±15.7	583.4±8.6	588.9±24.3	590.5±23.7	**580.4**±4.1
	599.6	**577.0**	**577.0**	**577.0**	**577.0**	**577.0**	**577.0**	**577.0**	**577.0**	**577.0**
ED-10-03	625.0±10.5	554.2±72.4	571.0±80.7	520.3±30.9	**513.6**±1.4	514.1±2.3	514.1±2.3	514.9±3.0	513.9±1.9	514.5±2.6
	608.0	**513.4**	**513.4**	**513.4**	**513.4**	**513.4**	**513.4**	**513.4**	**513.4**	**513.4**
ED-15-34	539.4±28.5	447.4±3.9	455.5±15.7	448.3±6.4	446.3±2.3	446.8±2.3	446.9±3.2	448.7±8.1	447.1±3.3	**446.0**±1.8
	511.5	**444.5**	**444.5**	**444.5**	**444.5**	**444.5**	**444.5**	**444.5**	**444.5**	**444.5**
ED-15-77	569.7±0.0	508.6±11.1	506.0±12.9	505.0±6.9	506.3±8.4	505.6±9.2	**502.8**±6.4	503.4±6.2	504.6±7.4	505.1±8.3
	569.7	**498.7**	**498.7**	**498.7**	**498.7**	**498.7**	**498.7**	**498.7**	**498.7**	**498.7**
ED-10-16	718.2±18.8	591.5±2.7	590.2±2.1	590.8±6.1	588.9±2.8	590.0±5.6	589.2±2.6	590.1±3.3	**588.8**±2.1	590.0±4.8
	662.5	**585.1**	586.7	**585.1**	**585.1**	586.7	586.7	586.7	586.7	586.7
ED-15-54	479.5±0.0	422.0±10.5	421.9±9.2	416.8±5.3	417.7±8.3	426.2±11.6	416.6±5.9	**414.6**±3.3	416.0±4.4	414.8±3.3
	479.5	**410.5**	**410.5**	**410.5**	**410.5**	**410.5**	**410.5**	**410.5**	**410.5**	**410.5**
ED-10-17	796.0±11.1	700.6±21.7	702.5±26.3	707.2±28.8	693.9±18.7	703.8±26.5	696.0±16.9	**691.6**±11.1	692.1±17.7	692.8±18.2
	783.7	**683.8**	**683.8**	**683.8**	**683.8**	**683.8**	**683.8**	**683.8**	**683.8**	**683.8**
ED-10-14	876.2±19.8	779.8±63.3	810.3±81.7	764.4±36.4	753.2±5.5	755.3±17.2	752.5±5.0	752.1±7.0	750.9±6.6	**749.9**±6.3
	821.5	**744.6**	**744.6**	**744.6**	**744.6**	**744.6**	**744.6**	**744.6**	**744.6**	**744.6**
ED-15-11	570.0±0.0	510.5±18.7	504.2±17.7	496.4±13.7	495.5±10.4	493.5±8.8	492.3±8.6	492.9±9.0	497.1±16.7	**490.6**±8.6
	570.0	**484.0**	485.0	**484.0**	**484.0**	**484.0**	**484.0**	**484.0**	**484.0**	**484.0**
ED-15-30	693.0±0.5	600.8±22.9	602.5±24.4	598.1±22.9	581.4±19.8	579.0±19.1	**574.2**±15.8	602.1±25.6	600.5±22.9	601.0±17.2
	692.5	**560.5**	**560.5**	**560.5**	**560.5**	**560.5**	**560.5**	**560.5**	561.5	562.5
ED-15-31	679.5±0.0	610.9±10.8	612.5±11.3	606.7±13.5	607.5±9.7	604.0±12.4	**600.4**±11.8	607.4±9.3	607.0±9.8	609.1±7.7
	679.5	592.5	**589.5**	**589.5**	**589.5**	**589.5**	**589.5**	**589.5**	**589.5**	**589.5**
ED-15-35	782.4±42.8	679.8±4.3	679.6±4.9	681.6±7.2	**678.2**±3.6	680.4±7.3	678.6±3.5	679.7±5.8	680.8±6.4	679.1±5.8
	738.5	**674.5**	**674.5**	**674.5**	**674.5**	**674.5**	**674.5**	**674.5**	**674.5**	**674.5**
ED-15-15	881.0±0.0	749.0±13.5	750.4±14.9	746.8±15.6	739.0±11.1	**737.7**±9.9	738.7±10.9	740.0±10.5	742.5±10.1	744.7±10.0
	881.0	**728.0**	729.0	**728.0**	**728.0**	**728.0**	**728.0**	**728.0**	**728.0**	732.0
ED-15-16	731.0±0.0	693.2±13.1	695.4±12.0	690.9±12.1	**679.7**±9.6	683.1±12.6	682.3±11.4	685.6±11.3	687.6±13.4	686.7±11.4
	731.0	**673.0**	**673.0**	**673.0**	**673.0**	**673.0**	**673.0**	**673.0**	**673.0**	**673.0**
ED-15-10	873.5±0.0	671.4±5.9	672.0±5.7	671.4±7.0	676.8±9.9	674.5±8.0	674.1±9.6	671.0±6.1	672.1±6.2	**669.8**±4.6
	873.5	**666.5**	**666.5**	**666.5**	**666.5**	**666.5**	**666.5**	**666.5**	**666.5**	**666.5**
ED-15-60	714.5±0.0	678.7±10.9	679.5±13.8	680.2±12.6	673.9±7.7	673.5±7.5	675.6±11.6	672.9±9.8	674.7±7.0	**672.3**±7.7
	714.5	**658.5**	**658.5**	659.5	659.5	**658.5**	**658.5**	**658.5**	659.5	**658.5**
ED-15-57	938.8±0.5	841.6±16.0	834.7±11.4	837.0±12.1	844.0±13.7	844.1±13.7	834.3±13.7	835.7±14.4	**833.0**±10.9	834.2±10.9
	938.5	823.5	822.5	**821.5**	822.5	**821.5**	**821.5**	**821.5**	823.5	822.5
ED-15-51	918.0±0.0	790.3±8.0	789.5±8.5	786.3±6.0	787.7±4.8	786.3±6.1	**785.1**±3.8	786.9±6.3	789.9±9.7	788.5±6.8
	918.0	779.0	**777.0**	778.0	779.0	779.0	779.0	778.0	**777.0**	778.0
ED-15-69	978.3±5.1	823.0±11.0	825.2±11.9	822.2±10.2	818.1±7.7	818.3±7.5	**815.9**±5.9	820.4±9.8	821.2±10.4	822.3±10.8
	973.0	**809.0**	**809.0**	810.0	**809.0**	**809.0**	**809.0**	**809.0**	**809.0**	**809.0**
ED-15-26	1099.0±0.0	856.7±16.8	855.0±20.4	858.9±24.0	849.8±13.0	**847.9**±6.8	852.1±15.4	854.0±17.4	850.0±8.1	851.6±11.4
	1099.0	845.0	**844.0**	**844.0**	**844.0**	**844.0**	845.0	**844.0**	**844.0**	**844.0**
ED-15-19	1177.5±63.1	1018.6±15.9	1018.6±15.7	1017.8±16.4	1010.4±8.9	1010.9±11.0	**1005.6**±4.1	1016.8±15.6	1021.5±16.9	1016.3±15.3
	1115.5	**999.5**	**999.5**	**999.5**	**999.5**	**999.5**	**999.5**	**999.5**	**999.5**	**999.5**
ED-15-39	1227.0±9.2	939.2±7.7	939.6±8.5	939.1±10.3	932.7±7.9	932.3±8.7	**929.6**±7.3	940.2±5.3	939.1±6.3	940.9±6.6
	1218.0	927.0	924.0	923.0	**922.0**	**922.0**	**922.0**	928.0	929.0	926.0
ED-15-24	1280.5±0.0	1053.6±30.6	1050.6±29.7	1044.4±27.5	1044.8±24.2	1046.3±23.5	**1032.5**±9.2	1041.0±22.9	1051.7±31.0	1044.1±22.9
	1280.5	1029.5	**1028.5**	**1028.5**	**1028.5**	**1028.5**	**1028.5**	**1028.5**	**1028.5**	**1028.5**
ED-15-13	1699.5±0.0	1406.5±69.7	1424.9±101.8	1392.6±7.9	1410.2±34.0	1402.7±27.5	1394.3±18.5	**1390.1**±6.0	1394.0±8.6	1391.8±6.5
	1699.5	1382.5	1382.5	**1381.5**	**1381.5**	**1381.5**	**1381.5**	**1381.5**	**1381.5**	**1381.5**
ED-15-27	1357.0±2.5	1233.6±21.3	1243.2±28.2	1233.5±19.6	1237.5±18.7	1243.3±21.7	1231.5±16.3	**1230.9**±13.2	1234.2±22.7	1234.4±15.9
	1354.5	**1220.5**	**1220.5**	**1220.5**	**1220.5**	**1220.5**	**1220.5**	**1220.5**	**1220.5**	**1220.5**
ED-15-21	1511.5±0.0	1291.3±25.9	1287.7±24.7	1285.8±25.0	1269.0±19.5	1273.2±22.1	**1259.3**±13.2	1287.2±26.1	1295.5±26.4	1287.8±27.1
	1511.5	1245.5	1252.5	1248.5	**1246.5**	**1246.5**	**1246.5**	**1246.5**	1248.5	1247.5
ED-15-08	1462.5±0.0	1333.0±29.5	1336.2±32.9	1325.6±27.8	1326.7±22.2	1330.7±33.9	**1309.5**±5.9	1328.6±24.7	1322.5±23.5	1337.4±45.7
	1462.5	**1304.5**	**1304.5**	**1304.5**	**1304.5**	**1304.5**	**1304.5**	**1304.5**	**1304.5**	**1304.5**
ED-14-02	2025.7±0.0	2012.5±21.5	2022.6±9.3	**1983.5**±18.9	1994.8±31.3	2005.3±30.6	2002.8±25.9	2014.4±24.2	2017.6±21.6	2012.7±21.9
	2025.7	1971.6	1973.4	1971.6	**1953.2**	**1953.2**	**1953.2**	1972.5	1971.6	1967.2
ML-100-1	2082.8±37.9	1890.3±18.8	1900.9±22.0	**1877.9**±12.5	1908.4±39.2	1912.3±34.8	1878.9±23.3	1900.0±33.4	1907.3±34.6	1897.3±28.0
	2053.3	1870.7	1870.7	**1869.6**	**1869.6**	**1869.6**	**1869.6**	**1869.6**	**1869.6**	**1869.6**
ED-15-28	1868.2±0.0	1685.8±14.6	1689.8±18.5	1684.4±18.7	**1668.7**±9.2	1671.8±10.4	1669.8±13.1	1674.2±16.0	1673.8±12.5	1670.6±13.1
	1868.2	1664.6	1657.8	1657.0	1654.8	1654.2	**1653.0**	1654.8	1654.8	1656.2
ED-11-03	1756.5±0.0	1561.9±49.6	1562.4±39.1	1555.1±28.3	1546.4±23.8	1543.6±21.1	**1533.7**±12.8	1557.1±30.3	1553.3±26.1	1551.2±23.4
	1756.5	1519.5	1519.5	1520.5	1519.5	1519.5	**1518.5**	1528.5	1528.5	1530.5
ED-15-29	1640.0±1.0	1444.7±26.2	1437.6±23.5	1438.1±28.3	**1433.0**±21.2	1434.9±20.9	1440.1±23.6	1443.5±25.0	1440.6±28.9	1438.0±26.4
	1639.0	1411.0	**1410.0**	**1410.0**	**1410.0**	**1410.0**	**1410.0**	**1410.0**	1411.0	**1410.0**
ED-15-07	1798.0±0.0	1461.6±12.7	1461.3±13.2	1464.6±15.5	1456.9±8.6	1464.8±15.8	**1455.5**±9.3	1460.5±12.8	1459.1±14.0	1457.5±9.4
	1798.0	1446.0	**1443.0**	1446.0	1445.0	1448.0	**1443.0**	**1443.0**	**1443.0**	1446.0
ED-15-22	2093.5±0.0	1781.1±48.9	1769.3±47.0	1770.2±47.8	1752.1±24.1	1756.0±34.0	**1750.2**±19.2	1780.9±54.8	1768.6±54.2	1775.6±54.4
	2093.5	1721.5	1723.5	1725.5	**1720.5**	**1720.5**	**1720.5**	1725.5	1723.5	1725.5
ED-15-09	2033.0±0.0	1731.7±48.4	1720.4±43.2	1715.7±36.8	1717.7±40.1	1722.7±36.8	1718.6±30.8	1723.0±36.2	**1714.7**±28.1	1716.6±38.7
	2033.0	1685.0	1685.0	1685.0	**1684.0**	1686.0	**1684.0**	**1684.0**	1685.0	1685.0
ED-15-20	2633.1±48.0	2410.4±18.5	2412.2±17.5	2415.3±18.7	2416.1±15.3	2415.4±15.7	2422.7±15.4	2411.2±15.4	**2407.8**±18.3	2417.6±20.5
	2607.5	2382.5	2384.5	2384.5	**2382.5**	2390.1	**2382.5**	2384.5	**2382.5**	**2382.5**
ED-15-17	2615.0±0.0	2366.0±49.3	2360.6±46.3	2366.9±43.0	2340.1±30.1	2339.4±31.8	**2319.3**±18.2	2356.5±43.6	2350.6±36.5	2353.8±39.5
	2615.0	2307.0	2313.0	2305.0	2306.0	2308.0	**2304.0**	2308.0	2308.0	2308.0
ED-15-33	2571.5±0.0	2337.2±16.3	2333.4±17.5	2331.3±20.0	2329.4±16.0	2324.1±16.0	**2322.5**±22.3	2342.3±18.0	2345.7±18.9	2345.6±14.5
	2571.5	**2298.5**	**2298.5**	**2298.5**	**2298.5**	**2298.5**	**2298.5**	2306.5	2316.5	2316.5
ED-15-40	3033.0±0.0	2551.6±55.7	2556.4±54.5	2526.9±16.9	2527.1±17.6	2535.9±41.4	**2518.4**±4.5	2531.6±26.4	2527.3±17.0	2530.2±18.6
	3033.0	**2512.0**	**2512.0**	**2512.0**	**2512.0**	**2512.0**	**2512.0**	**2512.0**	**2512.0**	**2512.0**
ED-15-23	3385.0±0.0	2896.3±23.0	2893.0±24.5	2899.2±26.1	2875.1±21.2	2879.3±16.8	**2873.3**±17.0	2891.6±20.0	2889.2±16.5	2889.7±27.2
	3385.0	2856.0	2842.0	2868.0	2846.0	2847.0	2842.0	2843.0	2860.0	**2840.0**
ML-150-2	5072.7±63.5	4733.1±58.2	4718.0±56.0	4704.2±47.6	4710.4±66.7	4710.8±61.4	**4700.8**±56.3	4713.0±59.9	4737.0±52.3	4708.8±56.1
	4950.5	**4639.8**	**4639.8**	**4639.8**	**4639.8**	**4639.8**	**4639.8**	**4639.8**	**4639.8**	**4639.8**
ED-15-32	3646.3±2.1	3159.8±31.9	3160.7±32.0	3156.0±29.7	**3135.7**±24.6	3139.0±26.7	3144.1±30.2	3148.4±35.4	3160.8±33.6	3157.6±36.4
	3642.5	**3112.5**	**3112.5**	3113.5	**3112.5**	**3112.5**	**3112.5**	**3112.5**	**3112.5**	**3112.5**
ED-15-14	4064.5±0.0	3573.1±27.3	3571.8±25.5	3566.9±33.0	3552.2±27.9	**3545.7**±29.7	3559.5±29.6	3547.9±28.5	3558.8±35.0	3556.2±33.2
	4064.5	3513.5	3513.5	3507.5	**3505.5**	3508.5	3511.5	3508.5	3512.5	3510.5
ED-10-50	6414.4±14.1	5531.9±32.7	5530.6±37.1	5531.9±42.9	5519.0±42.6	**5514.8**±44.0	5540.4±54.6	5548.1±76.9	5535.2±53.4	5561.9±67.4
	6392.2	5488.8	5482.2	5487.8	5477.5	5473.5	5475.2	**5469.8**	5478.2	5479.2
ML-200-3	8855.2±11.7	8564.2±103.5	8610.4±85.2	**8451.8**±66.3	8553.6±99.2	8574.1±102.7	8483.7±88.6	8568.0±108.6	8534.2±107.8	8490.0±86.9
	8834.4	8426.2	**8421.7**	**8421.7**	**8421.7**	**8421.7**	**8421.7**	**8421.7**	**8421.7**	**8421.7**
ED-11-01	6136.8±0.0	5616.3±34.5	5616.3±30.0	5612.5±35.9	5608.0±25.7	5602.8±23.1	5593.0±22.4	5587.0±21.1	5583.3±16.0	**5579.3**±17.8
	6136.8	5563.8	5572.2	5567.6	5565.0	5566.8	5563.2	**5560.0**	5562.2	**5560.0**
ED-11-02	12695.6±0.0	10979.8±56.2	10978.3±75.0	10983.0±64.8	**10959.8**±51.7	11015.0±80.9	10973.3±64.5	10980.1±61.9	10971.0±57.7	10968.9±51.4
	12695.6	10913.6	10913.4	**10919.0**	10919.2	10922.8	10923.0	10919.2	**10919.0**	**10919.0**
ML-250-4	13204.3±81.4	13080.5±338.0	13229.1±0.0	**12454.7**±309.1	12954.1±427.4	12950.5±433.1	12741.9±463.9	13050.2±364.0	13077.6±344.5	12870.4±446.9
	12886.4	12337.4	13229.1	12287.2	**12281.8**	**12281.8**	**12281.8**	12319.6	12287.2	12293.2
ML-250-5	13159.6±82.1	13204.7±299.8	13173.4±335.6	**12489.3**±370.3	12939.0±486.7	12898.5±504.3	12611.6±461.1	13041.7±440.7	13104.9±403.1	12942.5±481.7
	12964.9	12314.4	12332.1	12271.8	**12257.5**	**12257.5**	**12257.5**	12262.1	12267.1	12271.8

Now we analyze the three ways of computing the initial solution and the three methods used to generate the descendant population. Concerning the initial solution, the statistical analysis rejects the hypothesis of all methods having the same performance (p-value = 1.1e-9). Figure 3.(b) shows the ranking obtained by the Friedman test. It can be observed that the Borda algorithm is significantly better than Rd and BPA at generating the initial solution. Regarding the method for generating the descendant population, a similar analysis reports that there is at least one method with a different performance (p-value = 5.7e-5), and Figure 3.(c) shows the ranking obtained by the Friedman test. As we can observe, Uni is significantly better than All and Rd as a method for generating the descendant population.

Overall, we conclude that the evolution strategy Borda-Uni is the best option, obtaining (on average) an improvement of 35% over BPA and 12.5% over the state-of-the-art algorithm LIA_G^{MP2}.

4.2.2 Efficiency results

In this section we analyze the efficiency of the algorithms from two points of view:

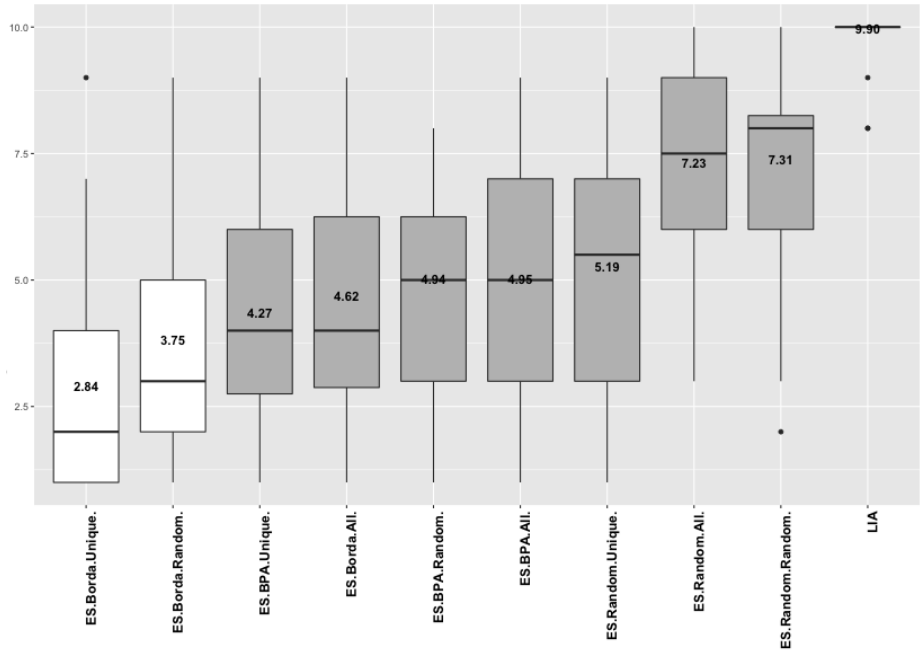
- CPU time of each *complete* run ($10n^2$ evaluations).
- number of evaluations *until the best solution in the run is found.*

Table 3 shows these two values for each algorithm/dataset averaged over the 30 runs.

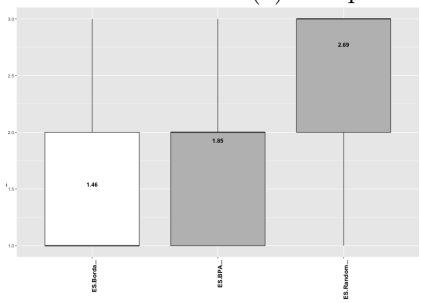
By applying the same statistical analysis procedure as for accuracy, we observe that the three ES using Borda to generate the initial solution are significantly better than the rest (Figure 4). Therefore, the ES configurations using Borda as the initialization method not only are the best choice from an accuracy point of view, but also need less evaluations to converge towards better solutions. Regarding the method for generating the descendant population, the Friedman test does not reject the null hypothesis of all methods being equivalent (p-value = 0.1461).

As regards the CPU time, note that, though the different configurations of the ES algorithm are slower (by far) than LIA_G^{MP2}, they always obtain significantly better solutions (see Section 4.2.1). In any case, the ES configurations are clearly competitive: Borda-Uni needs less than one second when $n \leq 75$, less than one minute when $n \leq 200$ and 2.5 minutes for $n = 250$. The statistical analysis also reports that the ES configurations using Borda as initialization method require significantly more CPU time than the other combinations (Figure 5). This can be easily explained: since the complexity of most of the mutation operators depends on the number of buckets, those methods starting from solutions with a larger number of buckets (Borda[6]) require more CPU time, at least in the first generations.

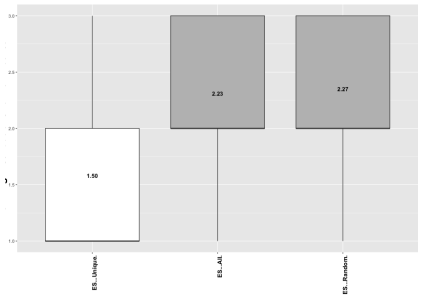
[6]In general, the solutions returned by Borda (allowing ties) have a number of buckets similar to the number of items, because ties are rarely found.



(a) Comparison for the ten methods.



(b) Comparison by initialization.



(c) Comparison by strategy.

Figure 3: Distribution of Friedman-based rank for accuracy (fitness).

Table 3: Number of evaluated solutions ($\times 10^3$) until the best one is found (first row) and CPU time in seconds (second row). Average $\pm$ variance over the 30 runs.

Dataset	LIA	Rd-Rd	Rd-All	Rd-Uni	Borda-Rd	Borda-All	Borda-Uni	BPA-Rd	BPA-All	BPA-uni
ED-10-11	-	19±6	18±6	15±7	15±7	14±7	16±6	16±7	15±7	16±7
	0.0014±0.0032	0.20±0.03	0.20±0.04	0.20±0.03	0.22±0.03	0.22±0.03	0.23±0.03	0.21±0.04	0.20±0.04	0.20±0.03
ML-050-6	-	16±6	19±5	18±5	15±8	16±8	18±5	16±7	15±7	16±7
	0.0006±0.0005	0.21±0.01	0.21±0.01	0.22±0.01	0.23±0.01	0.24±0.02	0.25±0.01	0.22±0.02	0.21±0.01	0.21±0.01
ED-10-02	-	19±6	19±5	18±7	15±7	16±6	18±6	17±6	17±8	15±6
	0.0002±0.0004	0.23±0.02	0.23±0.02	0.23±0.02	0.24±0.02	0.25±0.02	0.24±0.01	0.23±0.03	0.23±0.02	0.22±0.01
ED-10-15	-	16±7	16±8	14±7	15±7	14±8	15±6	15±7	14±6	17±7
	0.0001±0.0003	0.23±0.00	0.22±0.00	0.23±0.00	0.25±0.01	0.25±0.01	0.25±0.01	0.23±0.01	0.22±0.00	0.23±0.00
ED-10-03	-	16±7	17±6	15±7	18±7	14±7	16±6	16±5	14±6	17±7
	0.0002±0.0004	0.27±0.02	0.26±0.01	0.27±0.01	0.29±0.02	0.29±0.00	0.29±0.00	0.27±0.01	0.26±0.00	0.27±0.01
ED-15-34	-	19±8	21±7	19±8	12±8	13±8	14±8	18±8	17±7	19±8
	0.0002±0.0005	0.29±0.01	0.29±0.01	0.31±0.02	0.33±0.02	0.31±0.01	0.33±0.01	0.30±0.02	0.29±0.01	0.30±0.01
ED-15-77	-	27±4	26±5	25±6	23±7	21±9	21±7	23±7	23±7	25±6
	0.0001±0.0003	0.31±0.01	0.31±0.01	0.32±0.02	0.33±0.02	0.32±0.01	0.33±0.01	0.31±0.01	0.30±0.01	0.31±0.01
ED-10-16	-	16±7	17±6	17±8	23±8	22±9	21±6	14±7	15±7	18±8
	0.0001±0.0003	0.31±0.00	0.31±0.01	0.32±0.02	0.35±0.02	0.34±0.00	0.34±0.00	0.31±0.00	0.31±0.00	0.32±0.02
ED-15-54	-	28±8	26±7	26±8	24±10	22±9	25±9	23±8	27±7	24±8
	0.0001±0.0003	0.45±0.01	0.45±0.02	0.47±0.02	0.49±0.00	0.49±0.01	0.50±0.01	0.45±0.01	0.45±0.01	0.47±0.03
ED-10-17	-	22±11	23±8	23±10	17±10	25±10	24±11	21±13	17±11	19±11
	0.0001±0.0003	0.40±0.01	0.40±0.01	0.41±0.01	0.44±0.01	0.44±0.01	0.45±0.00	0.41±0.00	0.40±0.01	0.42±0.02
ED-10-14	-	22±12	27±9	26±12	20±10	19±8	22±11	24±10	19±12	19±10
	0.0001±0.0003	0.45±0.03	0.46±0.04	0.45±0.02	0.48±0.02	0.48±0.01	0.49±0.01	0.43±0.01	0.43±0.01	0.46±0.03
ED-15-11	-	34±5	32±6	33±5	25±11	26±10	24±10	27±9	27±9	26±9
	0.0001±0.0003	0.52±0.02	0.51±0.01	0.53±0.03	0.54±0.02	0.54±0.01	0.56±0.02	0.51±0.02	0.52±0.02	0.52±0.02
ED-15-30	-	33±7	33±6	33±6	24±9	24±11	24±10	28±10	31±9	28±12
	0.0002±0.0005	0.54±0.02	0.54±0.01	0.57±0.02	0.60±0.02	0.58±0.01	0.60±0.01	0.55±0.01	0.54±0.01	0.55±0.01
ED-15-31	-	35±8	33±7	34±7	29±12	30±12	30±10	33±9	30±7	33±10
	0.0001±0.0003	0.64±0.02	0.64±0.02	0.66±0.02	0.70±0.03	0.69±0.02	0.70±0.02	0.64±0.01	0.64±0.01	0.64±0.02
ED-15-35	-	30±12	30±10	28±10	13±9	21±13	20±13	24±11	30±12	26±12
	0.0001±0.0003	0.65±0.02	0.65±0.01	0.67±0.03	0.69±0.02	0.68±0.01	0.70±0.02	0.64±0.01	0.65±0.02	0.66±0.02
ED-15-15	-	39±8	37±8	38±8	29±13	21±12	29±11	35±9	32±10	34±9
	0.0002±0.0005	0.72±0.02	0.71±0.01	0.73±0.02	0.77±0.02	0.76±0.01	0.78±0.01	0.70±0.01	0.71±0.02	0.72±0.02
ED-15-16	-	35±11	33±10	33±12	23±13	25±13	19±9	33±11	34±10	31±12
	0.0001±0.0003	0.76±0.02	0.75±0.01	0.78±0.03	0.82±0.02	0.81±0.02	0.83±0.02	0.75±0.02	0.77±0.02	0.76±0.03
ED-15-10	-	34±9	37±9	39±10	32±13	34±13	30±15	35±12	33±12	33±12
	0.0001±0.0003	0.81±0.02	0.81±0.02	0.83±0.04	0.88±0.03	0.86±0.02	0.89±0.02	0.79±0.02	0.82±0.03	0.81±0.02
ED-15-60	-	39±10	42±9	43±9	26±14	29±15	28±14	38±9	36±12	35±12
	0.0001±0.0003	0.90±0.03	0.90±0.03	0.91±0.03	0.93±0.02	0.93±0.03	0.95±0.04	0.87±0.02	0.89±0.03	0.87±0.02
ED-15-57	-	42±9	40±9	43±9	36±11	28±15	37±13	41±8	39±13	39±12
	0.0001±0.0003	0.90±0.02	0.89±0.03	0.91±0.03	0.96±0.03	0.95±0.02	0.96±0.02	0.87±0.02	0.89±0.03	0.89±0.03
ED-15-51	-	44±10	46±10	45±12	33±15	36±16	34±15	46±8	47±10	44±13
	0.0002±0.0004	1.13±0.04	1.11±0.03	1.14±0.04	1.19±0.04	1.17±0.03	1.20±0.04	1.12±0.04	1.12±0.03	1.13±0.04
ED-15-69	-	52±11	49±15	46±14	37±18	41±14	41±16	49±13	46±17	48±16
	0.0002±0.0004	1.35±0.02	1.36±0.03	1.40±0.05	1.44±0.05	1.43±0.05	1.45±0.05	1.34±0.03	1.35±0.04	1.36±0.03
ED-15-26	-	51±14	48±12	44±12	38±17	44±16	42±16	48±15	45±16	47±14
	0.0001±0.0003	1.51±0.03	1.53±0.03	1.54±0.03	1.57±0.03	1.58±0.03	1.59±0.03	1.50±0.03	1.52±0.03	1.53±0.04
ED-15-19	-	59±14	59±14	59±15	47±18	52±18	40±18	55±17	58±10	48±16
	0.0001±0.0003	1.79±0.03	1.84±0.04	1.85±0.03	1.94±0.04	1.84±0.03	1.86±0.03	1.78±0.03	1.82±0.04	1.78±0.03
ED-15-39	-	63±14	67±11	66±12	54±19	51±19	47±20	61±14	61±13	64±13
	0.0001±0.0003	2.00±0.05	2.02±0.05	2.06±0.03	2.12±0.04	2.05±0.03	2.11±0.03	2.01±0.04	2.00±0.04	2.03±0.05
ED-15-24	-	54±20	53±17	43±21	50±23	51±24	42±21	50±23	54±22	50±21
	0.0002±0.0005	2.16±0.05	2.12±0.03	2.21±0.05	2.26±0.04	2.28±0.04	2.31±0.03	2.15±0.04	2.18±0.04	2.15±0.05
ED-15-13	-	56±21	52±22	52±19	51±23	60±20	49±18	50±22	52±20	47±19
	0.0002±0.0004	2.41±0.04	2.34±0.04	2.48±0.06	2.60±0.05	2.58±0.05	2.62±0.05	2.40±0.05	2.36±0.04	2.37±0.04
ED-15-27	-	71±16	71±15	68±14	57±26	62±24	67±19	67±21	56±22	64±19
	0.0003±0.0005	2.57±0.05	2.50±0.03	2.59±0.05	2.71±0.05	2.67±0.06	2.76±0.07	2.54±0.04	2.56±0.04	2.56±0.06
ED-15-21	-	79±11	80±8	78±14	70±20	72±18	70±17	69±18	69±15	72±13
	0.0002±0.0004	2.68±0.04	2.67±0.03	2.78±0.05	2.87±0.04	2.89±0.06	2.91±0.06	2.67±0.04	2.71±0.05	2.73±0.06
ED-15-08	-	81±17	80±17	76±22	76±18	74±19	57±27	74±22	77±18	77±21
	0.0002±0.0004	3.03±0.06	3.04±0.07	3.10±0.07	3.22±0.06	3.22±0.07	3.36±0.06	3.03±0.08	3.06±0.08	3.04±0.09
ED-14-02	-	28±20	20±13	56±26	33±28	26±24	36±25	29±27	29±23	31±22
	0.0002±0.0004	3.09±0.05	3.21±0.07	3.26±0.10	3.58±0.08	3.53±0.07	3.57±0.07	3.16±0.07	3.21±0.08	3.16±0.08
ML-100-1	-	76±21	69±25	88±19	63±30	50±30	64±28	74±29	69±28	73±23
	0.0001±0.0003	3.18±0.05	3.21±0.06	3.24±0.06	3.46±0.05	3.41±0.04	3.45±0.04	3.17±0.04	3.24±0.06	3.22±0.04
ED-15-28	-	101±5	101±4	100±6	93±15	90±19	88±17	96±9	94±15	93±16
	0.0002±0.0004	3.87±0.14	3.92±0.11	3.86±0.13	4.02±0.13	3.99±0.11	4.18±0.14	3.99±0.15	4.04±0.16	3.86±0.13
ED-11-03	-	78±24	80±22	78±24	68±24	67±27	63±20	81±17	80±20	77±17
	0.0002±0.0004	3.42±0.07	3.45±0.08	3.43±0.06	3.72±0.09	3.68±0.09	3.78±0.08	3.43±0.06	3.42±0.07	3.57±0.09
ED-15-29	-	95±15	99±13	94±15	79±25	89±19	89±20	90±17	95±15	96±16
	0.0003±0.0005	4.04±0.08	4.14±0.10	4.16±0.11	4.29±0.08	4.33±0.12	4.49±0.13	4.22±0.10	4.09±0.10	3.97±0.09
ED-15-07	-	96±25	94±22	92±22	94±25	91±24	82±26	90±27	94±22	95±19
	0.0002±0.0005	4.89±0.09	4.74±0.08	4.82±0.06	5.09±0.09	4.90±0.06	5.17±0.10	4.79±0.07	4.85±0.08	4.58±0.07
ED-15-22	-	108±14	106±13	109±18	98±26	93±28	95±21	106±17	109±15	113±13
	0.0004±0.0005	5.25±0.15	5.20±0.10	5.34±0.19	5.57±0.16	5.59±0.15	5.67±0.22	5.27±0.11	5.20±0.12	5.35±0.11
ED-15-09	-	115±21	101±21	108±25	98±31	107±23	116±18	110±19	114±15	97±30
	0.0005±0.0005	5.79±0.08	5.75±0.12	5.79±0.08	6.07±0.09	6.08±0.11	6.09±0.11	5.67±0.08	5.78±0.12	5.84±0.07
ED-15-20	-	125±24	122±19	127±19	105±34	86±39	115±33	121±23	118±26	125±28
	0.0004±0.0005	7.09±0.10	7.11±0.09	7.15±0.12	7.64±0.16	7.54±0.16	7.83±0.20	7.08±0.12	7.08±0.13	6.92±0.14
ED-15-17	-	143±17	135±22	141±21	134±27	138±26	114±31	142±26	139±21	146±17
	0.0004±0.0005	8.35±0.14	8.32±0.13	8.32±0.12	8.89±0.23	8.87±0.12	8.93±0.14	8.30±0.11	8.27±0.15	8.29±0.13
ED-15-33	-	106±31	111±33	120±29	91±36	97±44	114±39	116±31	114±40	116±30
	0.0003±0.0005	8.82±0.15	8.87±0.16	8.84±0.14	9.32±0.13	9.33±0.18	9.57±0.12	8.60±0.14	8.66±0.16	8.74±0.14
ED-15-40	-	128±39	144±35	137±38	119±39	121±41	101±46	126±36	124±44	124±36
	0.0003±0.0004	9.41±0.15	9.46±0.20	9.52±0.16	10.01±0.19	10.11±0.23	10.16±0.16	9.49±0.18	9.42±0.16	9.40±0.12
ED-15-23	-	175±24	173±29	162±32	158±36	128±44	162±35	165±31	166±27	169±26
	0.0004±0.0006	13.42±0.13	13.33±0.14	13.42±0.14	13.98±0.14	13.96±0.23	14.20±0.16	13.19±0.15	13.37±0.12	13.26±0.15
ML-150-2	-	170±45	171±45	153±55	120±46	131±57	133±52	167±53	145±51	121±56
	0.0007±0.0005	16.16±0.39	16.34±0.41	16.07±0.36	17.22±0.39	17.38±0.48	17.50±0.38	16.29±0.40	16.07±0.27	15.98±0.38
ED-15-32	-	195±32	206±35	195±39	159±54	167±57	160±62	182±45	185±43	205±26
	0.0007±0.0005	17.21±0.15	17.17±0.15	17.31±0.16	17.88±0.19	17.80±0.19	18.15±0.20	17.07±0.11	17.06±0.14	17.12±0.15
ED-15-14	-	236±26	232±33	238±27	202±50	210±43	217±42	226±35	229±34	224±36
	0.0005±0.0005	22.55±0.30	22.31±0.23	22.79±0.25	23.71±0.31	23.90±0.33	24.00±0.30	22.44±0.23	22.40±0.32	22.45±0.23
ED-10-50	-	252±33	245±49	252±38	242±41	249±37	247±58	267±26	261±37	262±36
	0.0005±0.0005	27.12±1.52	27.07±1.60	27.58±1.67	28.04±0.89	28.17±0.82	28.95±1.57	29.31±3.53	28.72±2.48	30.06±3.53
ML-200-3	-	253±77	237±78	217±86	154±86	164±83	179±107	199±94	250±96	159±81
	0.0008±0.0004	49.71±0.50	49.58±0.42	50.19±0.48	53.76±0.75	53.54±0.79	54.46±0.84	49.62±0.65	49.59±0.46	49.65±0.66
ED-11-01	-	565±8	556±21	551±23	556±16	560±21	533±52	542±40	541±29	543±34
	0.0010±0.0002	122.37±1.79	123.10±1.62	123.80±2.20	125.70±1.74	125.24±2.11	126.94±1.87	124.16±2.47	124.52±2.67	125.59±2.20
ED-11-02	-	512±89	528±91	534±65	521±88	493±96	512±97	503±123	533±69	507±92
	0.0010±0.0002	106.57±1.31	106.25±1.56	106.87±1.39	111.65±1.87	111.49±1.76	112.29±1.89	105.56±1.74	105.79±1.81	106.13±1.37
ML-250-4	-	93±70	83±57	100±58	67±36	83±42	65±31	72±43	96±54	69±41
	0.0013±0.0005	123.46±1.43	123.95±0.48	121.05±1.32	139.37±2.80	139.53±3.02	139.19±3.09	122.41±1.54	122.64±1.49	122.01±1.84
ML-250-5	-	239±169	240±154	146±116	257±170	216±168	191±139	200±150	182±128	240±175
	0.0014±0.0005	126.47±2.89	126.51±3.04	121.65±3.19	140.95±4.82	140.02±5.04	138.47±4.64	124.46±3.74	125.00±3.40	124.54±4.16

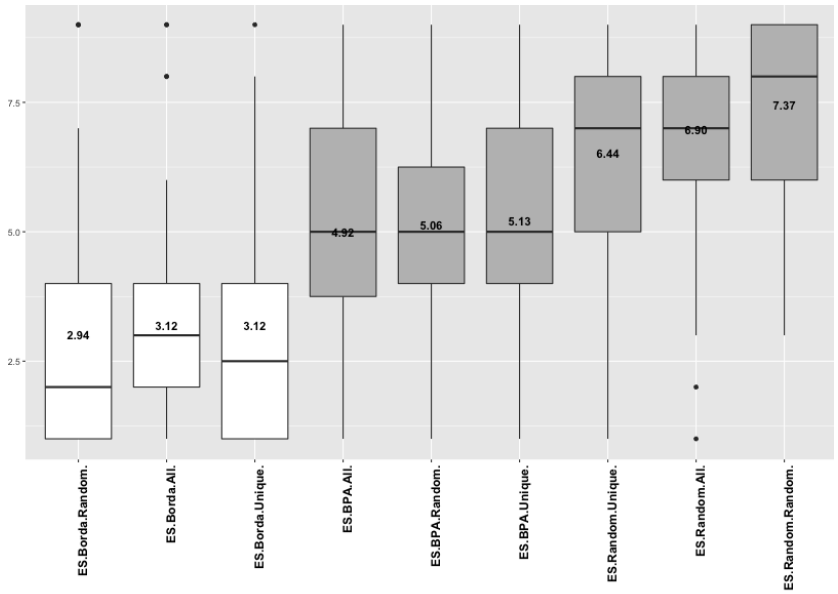


Figure 4: Distribution of Friedman-based rank for number of evaluations.

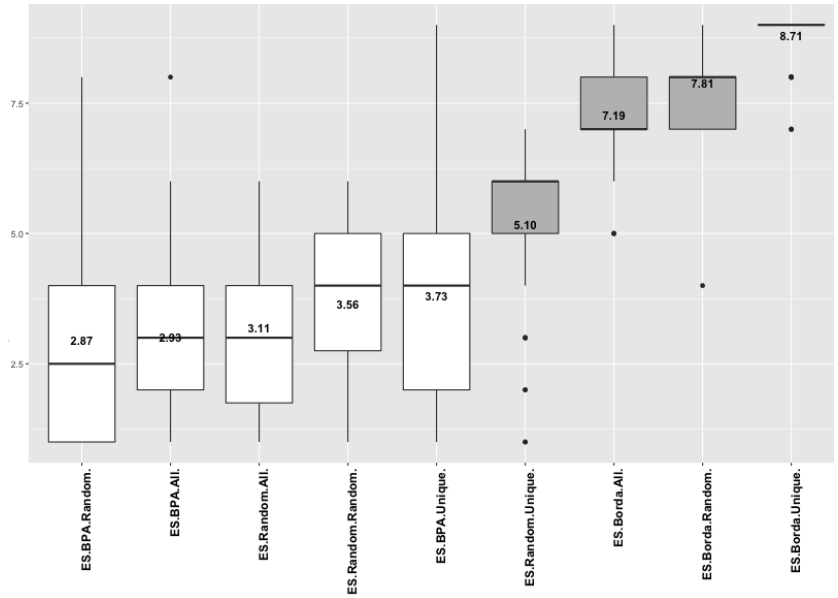


Figure 5: Distribution of Friedman-based rank for CPU time.

4.2.3 Inner structure of the solutions

In this subsection we analyze the number of buckets and the number of items included in the first bucket (the preferred ones) of the solutions obtained by the proposed algorithm and LIA_G^{MP2}. We show these data (averaged over the 30 runs) for each dataset and algorithm in Table 4.

From Table 4 we can draw the following conclusions:

- For the ED datasets, the number of buckets is around 10% the number of items. In the case of ML datasets this value drops to 4%.
- The number of buckets of the solutions obtained by the ES is greater than the number of buckets of the solutions obtained by LIA_G^{MP2}, by a factor of about 1.4.
- Regarding the configurations of the ES algorithm, the number of buckets of the solutions obtained is quite similar. In any case, Borda-Uni produces a slightly higher number of buckets (1.55 times more w.r.t. LIA_G^{MP2}), while the other combinations are in the interval 1.42-1.48. From the accuracy performance of the ES configurations, it may be inferred that this slight difference translates into a finer partition which leads to better solutions.
- The number of items placed in the first bucket, that is the preferred ones, is greater in the solutions obtained by LIA_G^{MP2} than in the solutions obtained by the ES configurations, by a factor of about 2.4. If we focus on the type of data (ED or ML), then LIA_G^{MP2} places fewer items in the first bucket for ML datasets (by a factor of 0.49) and more items for ED datasets (2.69).
- Regarding the ES algorithm, the difference in the number of items placed in the first bucket does not lie in the initialization method but in the way of generating the descendant population. Specifically, the method producing the fewest items in the first bucket is Uni. The difference between Uni and the rest of the strategies (Rd and All) is that Uni forces the use of bucket splitting in one out of seven generations, and this scheme of generating the new population seems to direct the search towards more promising regions.

To end this analysis, we compare the best solution obtained by each algorithm. In particular, we calculate the extended Kendall rank correlation coefficient τ_b [42] (4) and the rank correlation coefficient τ_X [42] (5) for all the pairs of these best bucket orders.

Given two bucket orders $\mathcal{B}^1$ and $\mathcal{B}^2$, the coefficient τ_b is

$$\tau_b(\mathcal{B}^1, \mathcal{B}^2) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}^1\alpha_{ij}^2}{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_{ij}^1)^2\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_{ij}^2)^2}} \tag{4}$$

Table 4: Number of buckets and number of items included in the first bucket

Dataset	LIA	Rd-Rd	Rd-All	Rd-Uni	Borda-Rd	Borda-All	Borda-Uni	BPA-Rd	BPA-All	BPA-uni
ED-10-11	3.6±0.7	3.3±0.7	3.2±0.8	2.9±0.9	3.5±1.4	3.0±1.3	3.3±1.5	3.3±1.1	3.0±0.8	3.5±1.3
	1.5±0.5	11.3±15.8	15.8±17.5	7.4±8.0	6.5±4.9	7.5±5.4	7.7±4.4	10.3±12.7	9.0±10.8	4.6±6.1
ML-050-6	3.0±0.0	4.3±0.6	4.7±1.1	4.8±0.8	4.7±0.9	4.8±0.9	5.5±0.9	4.9±1.1	4.6±0.8	4.9±1.5
	42.0±0.0	35.9±0.6	36.0±0.0	35.7±0.8	34.7±1.5	35.0±1.9	34.6±1.0	35.2±1.5	34.7±2.2	35.6±0.8
ED-10-02	4.8±0.7	6.5±2.8	6.8±2.8	6.4±2.2	5.1±1.8	5.6±2.0	5.8±1.4	5.0±2.0	5.8±2.9	4.8±1.7
	1.8±0.4	7.4±11.3	17.7±16.8	4.2±8.9	3.7±8.1	4.4±8.5	1.7±0.5	3.3±4.7	4.9±7.5	4.9±7.5
ED-10-15	3.1±0.3	3.1±0.3	3.0±0.2	3.0±0.2	3.0±0.2	2.9±0.3	2.9±0.3	3.0±0.2	2.9±0.3	3.0±0.0
	4.1±0.9	21.0±18.1	22.0±18.0	8.9±7.1	7.5±4.3	9.1±8.1	8.3±6.0	10.8±11.8	12.2±11.4	7.2±2.3
ED-10-03	5.5±0.5	3.8±0.4	3.9±0.4	3.8±0.4	4.0±0.2	3.9±0.3	3.9±0.3	3.8±0.4	3.9±0.3	3.8±0.4
	1.0±0.0	14.5±20.9	19.1±23.2	4.5±9.7	1.4±2.0	2.1±3.4	2.1±3.4	3.2±4.5	1.7±2.8	2.5±3.8
ED-15-34	4.3±0.5	6.3±1.3	5.9±1.2	6.5±1.0	6.6±1.0	6.3±0.9	7.7±1.3	6.1±1.1	6.1±0.9	5.9±1.0
	18.0±0.0	1.3±0.5	2.5±4.0	1.3±0.5	1.4±0.5	1.3±0.5	1.1±0.3	1.4±0.5	1.2±0.4	1.4±0.5
ED-15-77	3.0±0.0	5.9±0.4	5.7±0.5	5.8±0.4	6.0±0.5	5.9±0.5	6.0±0.2	5.9±0.4	5.8±0.5	5.9±0.5
	6.0±0.0	1.0±0.0	1.9±4.8	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-10-16	3.2±0.5	2.1±0.3	2.0±0.2	2.4±0.5	2.5±0.5	2.6±0.5	2.5±0.5	2.4±0.5	2.4±0.5	2.5±0.5
	4.4±6.8	18.3±4.2	19.2±3.1	12.8±8.3	10.8±8.5	10.3±8.3	11.9±8.4	13.4±8.2	13.1±8.4	10.8±8.5
ED-15-54	6.0±0.0	10.6±0.9	10.5±0.7	10.7±0.8	11.7±0.7	11.1±0.6	11.7±0.8	10.5±1.0	10.5±0.7	10.6±0.8
	5.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-10-17	2.1±0.3	2.3±0.6	2.2±0.7	2.5±0.7	2.3±0.7	2.6±0.9	2.5±0.6	2.6±0.5	2.6±0.6	2.7±0.6
	41.7±13.5	19.5±9.7	22.1±7.1	15.9±11.7	19.1±10.6	14.5±11.5	14.6±12.0	10.5±11.8	12.7±12.1	10.1±11.6
ED-10-14	3.7±0.7	3.2±1.7	3.9±1.9	2.9±0.9	2.6±0.5	2.6±0.6	2.5±0.5	2.6±0.5	2.4±0.5	2.4±0.5
	4.0±0.7	21.4±12.9	26.7±16.8	16.7±7.2	16.8±1.3	16.2±3.4	16.4±3.0	16.1±4.2	17.0±2.9	17.1±3.2
ED-15-11	5.0±0.0	8.9±1.0	8.8±1.1	8.9±1.0	9.2±1.2	9.2±0.7	10.0±1.5	8.5±1.0	8.8±1.1	8.7±0.8
	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-30	5.5±0.5	8.8±1.1	8.6±0.7	9.1±1.2	10.0±0.9	10.1±0.9	10.1±0.8	9.0±1.0	8.6±0.9	9.0±1.0
	2.0±0.0	1.7±0.5	1.7±0.5	1.9±0.3	2.0±0.0	2.0±0.0	2.0±0.0	1.4±0.5	1.3±0.5	1.4±0.5
ED-15-31	6.0±0.0	9.1±1.2	9.2±1.1	9.2±0.9	9.9±0.9	10.1±0.8	10.1±0.8	8.5±1.1	8.2±0.9	8.8±1.2
	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-35	3.2±0.4	5.2±0.7	5.2±0.8	5.0±0.6	4.9±0.9	4.9±0.7	5.0±0.7	4.8±0.8	5.2±0.8	5.2±1.0
	15.5±2.5	3.2±0.5	3.6±1.7	3.4±0.5	3.3±0.5	3.5±0.5	3.2±0.4	3.5±0.5	3.2±0.5	3.3±0.5
ED-15-15	6.0±0.0	9.7±1.1	9.4±1.1	10.1±1.3	10.0±0.9	9.6±0.8	10.6±0.9	9.1±0.9	8.8±0.9	9.2±1.0
	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-16	5.0±0.0	6.9±0.9	6.8±0.7	7.0±1.0	6.8±0.7	7.4±1.1	7.3±1.2	6.7±0.8	6.5±0.6	6.6±0.7
	2.0±0.0	2.0±0.2	2.0±0.0	1.9±0.3	2.0±0.0	2.0±0.0	2.0±0.0	2.0±0.0	1.9±0.3	1.9±0.3
ED-15-10	6.0±0.0	6.7±0.9	7.0±1.0	7.2±1.2	6.9±0.9	6.8±1.0	7.5±0.8	6.6±0.6	6.6±0.9	7.0±1.1
	4.0±0.0	2.7±1.5	3.2±1.4	2.3±1.5	3.4±1.2	3.6±1.0	1.9±1.4	3.7±0.9	3.6±1.0	3.2±1.4
ED-15-60	7.0±0.0	10.0±1.3	10.0±1.3	9.9±1.3	9.8±0.9	10.0±1.3	11.0±2.0	9.6±1.2	9.2±0.8	9.3±0.8
	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-57	6.7±0.5	9.8±1.0	9.8±0.9	10.2±1.0	10.7±0.7	10.7±0.7	10.5±0.7	9.2±0.9	9.7±1.2	9.7±1.2
	2.0±0.0	1.6±0.5	1.4±0.5	1.3±0.5	1.0±0.0	1.0±0.0	1.0±0.0	1.8±0.4	1.7±0.5	1.8±0.4
ED-15-51	6.0±0.0	8.4±1.0	8.5±1.1	8.6±1.0	8.4±0.8	8.8±1.2	8.9±1.1	8.8±1.1	8.6±1.0	9.1±1.2
	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-69	6.0±0.0	7.9±1.0	7.7±1.4	7.7±1.1	9.4±1.2	9.0±1.0	9.7±1.0	7.7±1.0	7.6±1.2	7.6±0.9
	4.0±0.0	2.0±0.8	2.2±0.5	1.9±0.6	1.7±0.5	1.4±0.5	1.6±0.5	2.0±0.6	2.0±0.5	1.9±0.4
ED-15-26	5.0±0.0	8.8±1.0	8.7±0.6	9.1±0.9	8.8±0.8	9.0±0.8	9.1±0.9	8.7±0.8	8.6±1.0	8.7±1.0
	1.0±0.0	1.9±1.4	1.3±0.9	1.1±0.6	1.0±0.0	1.1±0.6	1.0±0.0	1.9±1.4	2.2±1.5	2.0±1.4
ED-15-19	6.5±0.5	7.8±0.9	8.2±1.1	8.4±1.1	8.7±0.9	8.5±0.9	8.2±1.1	7.1±0.6	7.5±0.6	7.2±0.7
	4.0±0.0	1.6±0.8	1.6±0.8	1.3±0.7	1.1±0.4	1.1±0.4	1.0±0.2	2.6±0.8	2.0±1.0	2.6±0.8
ED-15-39	5.0±1.0	9.5±1.0	9.6±1.3	9.9±1.0	9.4±1.2	9.2±1.1	10.1±1.1	9.3±1.3	9.2±1.1	9.7±1.3
	7.0±0.0	1.4±0.5	1.5±0.5	1.2±0.4	1.2±0.4	1.4±0.5	1.1±0.4	1.4±0.5	1.4±0.5	1.2±0.4
ED-15-24	4.0±0.0	5.8±1.0	6.2±1.2	6.1±1.0	6.4±1.0	6.7±0.8	7.1±1.1	6.7±1.1	6.5±1.0	6.6±1.2
	22.0±0.0	12.0±3.5	8.6±4.4	8.3±4.1	6.5±5.1	6.0±5.6	2.6±3.6	9.4±5.5	6.7±5.4	8.3±5.6
ED-15-13	5.0±0.0	8.0±1.0	8.0±1.2	8.8±1.0	8.6±1.0	8.3±0.9	9.5±1.3	8.5±0.9	8.5±0.8	8.7±0.9
	2.0±0.0	3.4±1.3	5.7±5.8	1.8±0.5	1.8±0.4	1.9±0.3	1.3±0.5	1.7±0.5	1.5±0.5	1.8±0.5
ED-15-27	3.5±0.5	6.4±1.0	6.2±0.8	7.1±1.1	7.9±0.9	7.8±0.8	7.9±0.9	6.7±1.0	6.4±0.8	6.8±0.8
	26.0±0.0	4.4±5.7	5.2±6.9	2.6±4.2	1.2±0.8	1.1±0.6	1.2±0.8	2.5±1.4	5.0±6.6	2.3±1.4
ED-15-21	6.0±0.0	10.0±1.1	9.9±1.2	10.2±0.9	10.4±0.6	10.4±0.6	11.0±1.5	9.5±1.1	9.8±1.2	9.5±1.0
	4.0±0.0	2.0±0.7	2.1±0.8	1.7±0.7	1.3±0.5	1.7±0.5	1.5±0.5	2.4±0.8	2.5±0.8	2.2±0.9
ED-15-08	4.0±0.0	7.4±0.9	7.3±0.9	7.8±0.8	7.8±1.0	7.6±0.9	8.3±1.0	7.6±0.7	7.6±0.9	7.5±0.7
	13.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.2	1.0±0.0
ED-14-02	2.0±0.0	2.3±0.5	2.2±0.6	3.4±1.3	2.8±1.0	2.4±0.6	2.8±0.9	2.6±1.1	2.3±0.8	2.5±0.8
	5.0±0.0	73.2±38.4	91.0±16.3	19.3±34.0	53.4±41.3	66.7±38.6	55.5±44.7	72.9±37.7	78.7±33.7	73.1±38.1
ML-100-1	3.5±0.5	3.5±0.5	3.5±0.6	3.9±0.3	4.2±0.7	3.7±0.6	4.0±0.5	3.9±0.6	4.0±0.8	4.2±0.9
	3.5±0.5	32.8±27.8	38.1±26.0	12.4±16.9	19.1±15.5	30.7±22.4	10.0±13.7	26.3±23.3	28.8±23.0	19.4±23.1
ED-15-28	11.0±0.0	13.2±1.0	12.9±1.0	13.2±1.2	13.2±1.0	12.8±0.8	13.6±1.1	13.5±1.2	13.4±1.4	13.2±1.2
	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-11-03	7.0±0.0	8.1±1.2	8.1±1.2	8.2±0.8	8.9±0.8	8.9±0.8	8.9±0.7	8.2±0.9	8.2±0.9	8.1±0.9
	3.0±0.0	4.0±11.4	2.4±7.5	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-29	9.5±0.5	12.5±1.5	12.7±1.2	12.4±1.3	12.0±1.3	12.5±1.4	13.3±1.7	12.8±1.4	12.5±1.3	12.4±1.6
	2.5±1.5	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-07	6.0±0.0	9.7±1.2	9.2±1.1	10.2±1.4	11.1±1.1	10.8±0.9	11.1±1.4	8.9±0.9	9.1±1.0	9.1±0.8
	3.0±0.0	3.0±0.0	2.9±0.4	2.5±0.9	2.9±0.5	2.9±0.4	2.5±0.9	3.0±0.0	3.0±0.0	2.9±0.5
ED-15-22	8.0±0.0	12.5±1.8	12.7±1.8	12.8±2.0	12.2±1.9	12.5±2.4	14.5±2.2	12.3±2.0	12.7±1.9	12.6±1.8
	1.0±0.0	1.6±0.5	1.6±0.5	1.5±0.5	2.0±0.2	1.9±0.3	1.7±0.5	1.7±0.5	1.7±0.5	1.6±0.5
ED-15-09	9.0±0.0	8.8±0.9	8.9±0.8	9.0±0.9	9.2±1.1	9.3±1.1	9.6±1.1	9.0±0.8	8.8±1.0	8.8±0.7
	10.0±0.0	4.8±1.2	4.3±1.1	4.3±1.2	4.1±1.0	3.9±1.0	3.8±1.1	4.6±1.5	4.2±1.4	4.2±1.5
ED-15-20	5.9±0.5	10.3±1.1	10.3±0.9	10.8±1.0	10.8±1.2	10.4±1.0	12.1±1.8	10.3±0.8	9.9±1.0	10.5±1.4
	3.0±0.0	2.1±0.8	2.0±0.8	1.7±0.9	2.5±0.7	2.7±0.7	1.6±0.8	2.0±0.9	2.2±1.0	1.8±1.0
ED-15-17	6.0±0.0	10.3±1.0	10.2±0.8	10.3±0.6	11.2±1.1	11.0±0.9	11.3±1.0	10.1±0.8	10.3±1.0	10.5±1.1
	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
ED-15-33	6.0±0.0	8.2±1.1	8.7±1.1	9.2±1.4	8.9±1.0	9.2±1.2	9.9±1.7	8.2±1.1	8.4±0.9	8.6±1.1
	4.0±0.0	3.0±0.9	2.3±1.3	2.0±1.1	2.2±1.0	1.9±0.9	1.6±0.9	2.0±1.1	1.9±1.1	1.8±1.1
ED-15-40	4.0±0.0	7.9±1.0	7.9±1.4	8.0±0.8	8.5±1.1	8.1±1.0	8.3±1.2	8.1±0.7	8.1±0.8	8.3±1.1
	29.0±0.0	4.7±12.7	6.5±13.1	1.4±0.7	1.2±0.4	1.3±0.5	1.4±0.5	1.4±0.6	1.5±0.6	1.2±0.4
ED-15-23	7.0±0.0	12.2±1.5	12.2±1.3	12.1±1.4	12.0±1.1	11.5±1.4	13.5±1.4	11.5±1.2	11.5±1.2	11.8±1.3
	2.0±0.0	1.5±0.5	1.5±0.5	1.3±0.5	1.9±0.3	1.9±0.3	1.4±0.5	1.6±0.5	1.6±0.5	1.6±0.5
ML-150-2	4.5±0.6	5.0±1.6	5.3±1.7	5.2±1.7	4.7±1.4	5.4±1.7	5.4±1.5	5.1±1.6	5.1±1.4	5.1±1.6
	20.2±41.1	57.8±30.2	56.2±32.5	38.9±33.6	43.2±30.5	48.3±32.9	36.6±31.9	46.1±33.7	63.5±29.2	35.2±31.5
ED-15-32	5.8±0.4	8.4±0.7	8.4±0.8	8.4±0.8	8.1±0.5	8.2±0.5	8.8±1.2	8.2±0.7	8.3±0.7	8.4±0.7
	3.1±2.4	2.0±0.2	2.0±0.2	2.0±0.0	2.0±0.0	2.0±0.0	2.0±0.2	2.0±0.0	2.0±0.0	2.0±0.0
ED-15-14	7.0±0.0	12.8±1.6	12.4±1.3	12.7±1.4	13.0±1.5	13.1±1.4	13.5±1.6	12.5±1.2	12.6±1.3	12.6±1.1
	2.0±0.0	1.4±0.5	1.7±0.5	1.4±0.5	1.3±0.5	1.3±0.5	1.3±0.5	1.5±0.5	1.4±0.5	1.4±0.5
ED-10-50	17.0±2.0	13.0±4.2	13.3±4.9	13.8±5.4	13.5±4.1	13.3±3.9	13.6±3.8	18.1±9.3	17.4±8.2	19.0±9.7
	1.0±0.0	4.6±2.4	3.7±2.5	4.3±2.6	4.7±2.6	4.1±2.6	5.1±2.5	3.8±2.4	3.2±2.6	3.6±2.5
ML-200-3	4.0±0.0	3.7±0.9	3.4±0.7	4.6±0.7	4.1±0.9	3.9±0.8	4.4±0.7	3.9±0.9	4.2±0.9	4.4±0.6
	5.0±0.0	84.5±55.8	110.8±43.7	25.0±33.4	53.8±49.7	71.2±49.1	33.3±36.4	75.5±54.5	60.0±51.8	21.5±30.0
ED-11-01	19.0±0.0	25.1±1.3	26.2±1.9	25.8±1.8	26.0±1.6	25.2±1.7	25.7±1.6	25.8±1.9	26.0±2.0	26.7±1.4
	2.0±0.0	3.9±3.7	3.8±3.7	3.2±3.1	2.9±2.8	3.8±3.7	4.4±3.9	3.2±3.1	2.6±2.3	2.6±2.3
ED-11-02	3.0±0.0	10.5±1.6	10.9±2.2	10.5±1.5	10.1±2.1	10.0±1.4	10.4±2.4	10.4±1.9	10.2±1.9	10.8±1.2
	129.0±0.0	11.4±9.8	13.9±14.0	12.1±6.2	11.7±6.5	11.0±6.6	10.9±6.7	10.4±6.6	10.7±6.7	9.5±6.7
ML-250-4	2.3±0.5	2.8±0.4	2.9±0.3	2.1±0.4	2.6±0.5	2.6±0.5	2.4±0.5	2.7±0.5	2.8±0.4	2.5±0.5
	190.6±105.8	209.8±84.5	247.0±0.0	53.9±77.0	181.0±102.6	181.5±101.8	129.5±111.8	202.5±90.6	210.2±83.8	158.0±110.8
ML-250-5	5.9±0.7	4.9±1.6	5.0±1.6	2.6±1.5	4.2±1.9	4.0±1.9	3.1±1.7	4.2±1.9	4.5±1.8	4.2±2.0
	1.6±0.9	223.8±64.8	216.6±73.6	67.9±80.6	167.9±103.1	161.7±103.8	97.3±98.4	188.5±95.4	202.8±85.9	167.3±103.9

where

$$\alpha_{ij}^k = \begin{cases} 1 & \text{if object } i \text{ is ranked ahead of object } j \text{ in } \mathcal{B}^k \\ -1 & \text{if object } i \text{ is ranked behind object } j \text{ in } \mathcal{B}^k \\ 0 & \text{if } i \text{ and } j \text{ are tied in } \mathcal{B}^k, \text{ or if } i = j \end{cases}$$

On the other hand, the coefficient τ_X handles tied items in a different way. Given two bucket orders $\mathcal{B}^1$ and $\mathcal{B}^2$, the coefficient τ_X is

$$\tau_X(\mathcal{B}^1, \mathcal{B}^2) = \frac{\sum_{i=1}^n \sum_{j=1}^n \beta_{ij}^1 \beta_{ij}^2}{n(n-1)} \tag{5}$$

where

$$\beta_{ij}^k = \begin{cases} 1 & \text{if object } i \text{ is ranked ahead of or tied with object } j \text{ in } \mathcal{B}^k \\ -1 & \text{if object } i \text{ is ranked behind object } j \text{ in } \mathcal{B}^k \\ 0 & \text{if } i = j \end{cases}$$

Both τ_b and τ_X take values in the interval $[-1, 1]$; the closer they are to 1, the more similar the compared bucket orders are.

We have organized the data relating to these coefficients as follows. Let $\mathcal{B}^*(\mathrm{A}, \mathrm{D})$ be the best bucket order found by the algorithm A for the dataset D over the 30 runs. Then, given two algorithms A_1 and A_2, we denote by $\tau_b(\mathrm{A}_1, \mathrm{A}_2)$ (resp. $\tau_X(\mathrm{A}_1, \mathrm{A}_2)$) the average of the values $\tau_b(\mathcal{B}^*(\mathrm{A}_1, \mathrm{D}), \mathcal{B}^*(\mathrm{A}_2, \mathrm{D}))$ (resp. $\tau_X(\mathcal{B}^*(\mathrm{A}_1, \mathrm{D}), \mathcal{B}^*(\mathrm{A}_2, \mathrm{D}))$) for the 52 datasets considered in our study.

We show the results in Table 5. In this table, we have ordered the algorithms from top to bottom and from left to right, according to the order of the algorithms shown in Figure 6 (Friedman test ranking considering the performance of the algorithms). Then, above the main diagonal we show the coefficients τ_b, and below the main diagonal the coefficients τ_X.

	Borda-Uni	Borda-Rd	BPA-Uni	Borda-All	BPA-All	BPA-Rd	Rd-Uni	Rd-Rd	Rd-All	LIA_G^{MP2}	Borda	BPA
Borda-Uni	***	0.970	0.957	0.977	0.953	0.958	0.952	0.952	0.937	0.733	0.670	0.696
Borda-Rd	0.968	***	0.972	0.980	0.971	0.976	0.971	0.958	0.943	0.728	0.669	0.697
BPA-Uni	0.956	0.972	***	0.973	0.984	0.985	0.973	0.962	0.950	0.731	0.664	0.698
Borda-All	0.977	0.980	0.971	***	0.968	0.973	0.964	0.953	0.941	0.729	0.670	0.696
BPA-All	0.952	0.969	0.981	0.965	***	0.984	0.976	0.966	0.954	0.733	0.665	0.699
BPA-Rd	0.955	0.972	0.980	0.968	0.988	***	0.977	0.966	0.953	0.733	0.665	0.696
Rd-Uni	0.954	0.972	0.973	0.967	0.975	0.974	***	0.971	0.972	0.736	0.664	0.697
Rd-Rd	0.954	0.961	0.966	0.957	0.969	0.967	0.969	***	0.977	0.735	0.659	0.695
Rd-All	0.954	0.961	0.968	0.960	0.972	0.971	0.954	0.960	***	0.743	0.657	0.691
LIA_G^{MP2}	0.737	0.731	0.734	0.732	0.735	0.735	0.739	0.740	0.730	***	0.529	0.614
Borda	0.756	0.754	0.750	0.755	0.750	0.750	0.750	0.745	0.741	0.652	***	0.717
BPA	0.729	0.730	0.729	0.729	0.730	0.727	0.728	0.726	0.720	0.637	0.653	***

Table 5: Coefficients τ_b (above the main diagonal) and τ_X (below the main diagonal)

By studying Table 5 we can see that the solutions obtained by the ES configurations are quite similar: $\tau_b \geq 0.937$ and $\tau_X \geq 0.954$. When comparing the ES algorithm with LIA_G^{MP2}, both coefficients are around 0.73.

5 Conclusions

In this paper we propose the use of $(1 + \lambda)$ evolution strategies to tackle the optimal bucket order problem. Several configurations of the designed ES algorithm were tested using a benchmark of 52 datasets by combining different ways of generating the initial solution and the population of descendants. The statistical analysis carried out on the obtained results reveals that the ES algorithm performs better when it starts with an informed solution having a high number of buckets. In our study, this initial solution is obtained by using the Borda count method allowing ties. Our study also shows that Uni (the method that generates all the offspring in a generation by using the same mutation operator, which is cyclically changed for each iteration) is the best method for generating the population of descendants. The combination of these two strategies, Borda-Uni, proves to be the best configuration, obtaining a 35% improvement with respect to the standard greedy algorithms (BPA and Borda), and a 12.5% improvement with respect to the current state-of-the-art algorithm (LIA_G^{MP2}).

Our proposals are also very competitive in terms of efficiency. Even for the largest datasets considered ($n = 250$), the CPU time does not exceeds 2.5 minutes on a personal computer, which is a very reasonable response time for an evolutionary algorithm.

Acknowledgements

We would like to thank the reviewers for their valuable comments and suggestions. This study has been partially financed by the Spanish Government (MINECO) and FEDER funds through project TIN2016-77902-C3-01.

References

[1] S. Lin, Rank aggregation methods, Wiley Interdisciplinary Reviews: Computational Statistics 2 (5) (2010) 555–570.

[2] A. Ali, M. Meila, Experiments with Kemeny ranking: What works when?, Mathematical Social Sciences 64 (1) (2012) 28 – 40.

[3] J. L. Kemeny, J. G. Snell, Mathematical Models in the Social Sciences, Blaisdell, New York, 1962.

[4] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, E. Vee, Comparing and aggregating rankings with ties, in: Proceedings of the Twenty-third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2004, pp. 47–58.

[5] C. Dwork, R. Kumar, M. Naor, D. Sivakumar, Rank aggregation methods for the web, in: WWW, 2001, pp. 613–622.

[6] J. A. Aledo, J. A. Gámez, D. Molina, Using extension sets to aggregate partial rankings in a flexible setting, Applied Mathematics and Computation 290 (2016) 208 – 223.

[7] F. Franceschini, D. Maisano, L. Mastrogiacomo, A new proposal for fusing individual preference orderings by rank-ordered agents: A generalization of the Yager's algorithm, European Journal of Operational Research 249 (1) (2016) 209 – 223.

[8] E. M. García-Nové, J. Alcaraz, M. Landete, J. F. Monge, J. Puerto, Rank aggregation in cyclic sequences, Optimization Letters 11 (4) (2017) 667–678.

[9] A. Gionis, H. Mannila, K. Puolamaki, A. Ukkonen, Algorithms for discovering bucket orders from data, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06, 2006, pp. 561–566.

[10] S. Yasutake, K. Hatano, E. Takimoto, M. Takeda, Online rank aggregation, in: Proceedings of the Asian Conference on Machine Learning, Vol. 25 of Proceedings of Machine Learning Research, 2012, pp. 539–553.

[11] J. A. Aledo, J. A. Gámez, A. Rosete, Utopia in the solution of the bucket order problem, Decision Support Systems 97 (2017) 69–80.

[12] J. Ceberio, A. Mendiburu, J. A. Lozano, The linear ordering problem revisited, European Journal of Operational Research 241 (3) (2015) 686–696.

[13] N. Alon, Ranking tournaments, SIAM Journal on Discrete Mathematics 20 (1) (2006) 137142.

[14] N. Ailon, M. Charikar, A. Newman, Aggregating inconsistent information: Ranking and clustering, Journal of the ACM 55 (5) (2008) 23:1–23:27.

[15] W. D. Cook, Distance-based and ad hoc consensus models in ordinal preference ranking, European Journal of Operational Research 172 (2) (2006) 369 – 385.

[16] A. Ukkonen, K. Puolamaki, A. Gionis, H. Mannila, A randomized approximation algorithm for computing bucket orders, Information Processing Letters 109 (7) (2009) 356 – 359.

[17] S. Amodio, A. D'Ambrosio, R. Siciliano, Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach, European Journal of Operational Research 249 (2) (2016) 667 – 676.

[18] A. D'Ambrosio, G. Mazzeo, C. Iorio, R. Siciliano, A differential evolution algorithm for finding the median ranking under the Kemeny axiomatic approach, Computers & Operations Research 82 (2017) 126 – 138.

[19] A. D'Ambrosio, W. J. Heiser, A recursive partitioning method for the prediction of preference rankings based upon Kemeny distances, Psychometrika 81 (3) (2016) 774–794.

[20] M. Fortelius, A. Gionis, J. Jernvall, H. Mannila, Spectral ordering and biochronology of European fossil mammals, Paleobiology 32 (2) (2006) 206–214.

[21] U. Halekoh, W. Vach, A Bayesian approach to seriation problems in archaeology, Computational Statistics & Data Analysis 45 (3) (2004) 651 – 673.

[22] J. Feng, Q. Fang, W. Ng, Discovering bucket orders from full rankings, in: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, ACM, 2008, pp. 55–66.

[23] R. Agrawal, A. Halverson, K. Kenthapadi, N. Mishra, P. Tsaparas, Generating labels from clicks, in: Proceedings of the Second ACM International Conference on Web Search and Data Mining, WSDM '09, ACM, 2009, pp. 172–181.

[24] H. Beyer, D. Arnold, Theory of evolution strategies - A tutorial, in: Theoretical Aspects of Evolutionary Computing, Springer Berlin Heidelberg, 2001, pp. 109–133.

[25] Cook, Wade D., Kress, Moshe, Seiford, Lawrence M., An axiomatic approach to distance on partial orderings, RAIRO-Operations Research 20 (2) (1986) 115–122.

[26] S. Kenkre, A. Khan, V. Pandit, On discovering bucket orders from preference data, in: Proceedings of the 2011 SIAM International Conference on Data Mining, 2011, pp. 872–883.

[27] J. Bartholdi, C. A. Tovey, M. A. Trick, Voting schemes for which it can be difficult to tell who won the election, Social Choice and Welfare 6 (2) (1989) 157–165.

[28] N. Betzler, R. Bredereck, R. Niedermeier, Theoretical and empirical evaluation of data reduction for exact Kemeny rank aggregation, Autonomous Agents and Multi-Agent Systems 28 (5) (2014) 721–748.

[29] E. Hemaspaandra, H. Spakowski, J. Vogel, The complexity of Kemeny elections, Theor. Comput. Sci. 349 (3) (2005) 382–391.

[30] J. A. Aledo, J. A. Gámez, D. Molina, Tackling the rank aggregation problem with evolutionary algorithms, Applied Mathematics and Computation 222 (2013) 632–644.

[31] M. Mandal, S. Maity, A. Mukhopadhyay, Partial rank aggregation using multiobjective genetic algorithm: Application in ranking genes, in: Eighth International Conference on Advances in Pattern Recognition, ICAPR 2015, Kolkata, India, January 4-7, 2015, 2015, pp. 1–6.

[32] J. A. Aledo, J. A. Gámez, D. Molina, Using metaheuristic algorithms for parameter estimation in generalized Mallows models, Applied Soft Computing 38 (2016) 308–320.

[33] G. Napoles, R. Falcon, Z. Dikopoulou, E. Papageorgiou, R. Bello, K. Vanhoof, Weighted aggregation of partial rankings using ant colony optimization, Neurocomputing 250 (2017) 109–120.

[34] I. J. Good, The number of orderings of n candidates when ties are permitted, Fibonacci Quarterly 13 (1975) 11–18.

[35] E.-G. Talbi, Metaheuristics: From Design to Implementation, Wiley Publishing, 2009.

[36] P. Emerson, The original Borda count and partial voting, Social Choice and Welfare 40 (2) (2013) 353–358.

[37] P. Hansen, N. Mladenovi, Variable neighborhood search: Principles and applications, European Journal of Operational Research 130 (3) (2001) 449 – 467.

[38] N. Mattei, T. Walsh, Preflib: A library for preferences (http://www.preflib.org), in: Proceedings of the 3rd Conference on Algorithmic Decision Theory (ADT 2013), LNCS 8176, Springer, 2013, pp. 259–270.

[39] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[40] S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, Journal of Machine Learning Research 9 (2009) 2677–2694.

[41] J. Arias, J. Cózar, ExReport: Fast, reliable and elegant reproducible research (2015).
URL `http://exreport.jarias.es/`

[42] E. J. Emond, D. W. Mason, A new rank correlation coefficient with application to the consensus ranking problem, Journal of Multi-Criteria Decision Analysis 11 (1) (2002) 17–28.

A Statistical comparison including LIA_G^{MP2} as initialization method

Here we show the statistical analysis of accuracy for the ten algorithms studied in Section 4.2, the three ES using LIA_G^{MP2} to obtain the initial solution, and the three initialization methods considered in our study (Random, BPA and Borda count allowing ties). As can be observed in Figure 6 and Table 6, none of the algorithms using LIA_G^{MP2} to initialize the search is in the outstanding group.

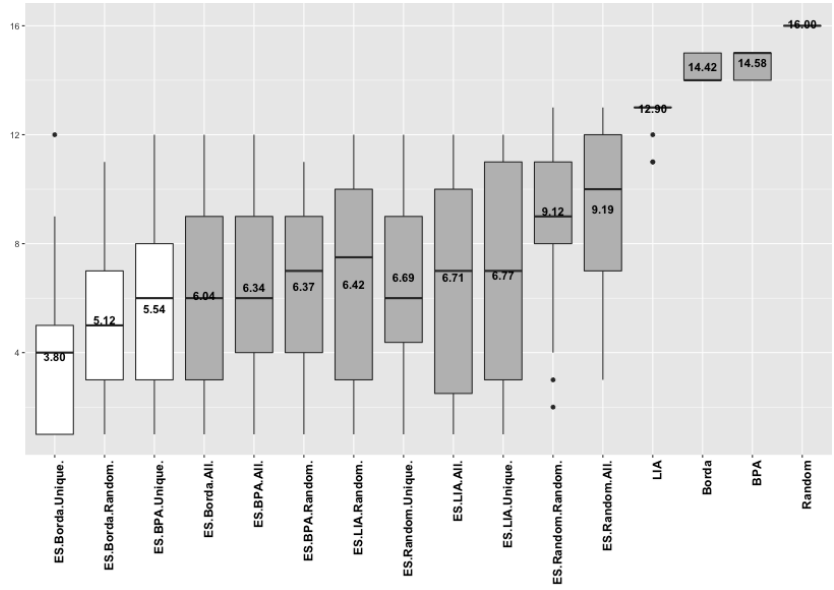


Figure 6: Distribution of Friedman-based rank for fitness (distance).

Table 6: Summary of control post-hoc test. Boldfaced results represent non-rejected hypotheses with respect to Borda-Uni ($\alpha = 0.05$).

method	rank	pvalue	win	tie	loss
Borda-Uni	3.80	-	-	-	-
Borda-Rd	5.12	**1.5829e-01**	33	0	19
BPA-Uni	5.54	**1.2465e-01**	33	0	19
Borda-All	6.04	4.9256e-02	41	1	10
BPA-All	6.34	2.9833e-02	34	0	18
BPA-Rd	6.37	2.9833e-02	41	0	11
LIA_G^{MP2}-Rd	6.42	2.9596e-02	35	0	17
Rd-Uni	6.69	1.4451e-02	41	0	11
LIA_G^{MP2}-All.	6.71	1.4451e-02	35	0	17
LIA_G^{MP2}-Uni	6.77	1.3158e-02	36	0	16
Rd-Rd	9.12	1.2346e-07	49	0	3
Rd-All	9.19	8.3517e-08	48	0	4
LIA_G^{MP2}	12.90	2.1629e-21	52	0	0
Borda	14.42	6.8802e-29	52	0	0
BPA	14.58	1.1052e-29	52	0	0
Random	16.00	7.4894e-38	52	0	0