

A Diversity-Aware Memetic Algorithm for the Linear Ordering Problem: Improving Best-Known Solutions for Standard Benchmarks

Lázaro Lugo

Centro de Investigación en Matemáticas <https://orcid.org/0000-0002-5134-7739>

Carlos Segura (✉ carlos.segura@cimat.mx)

Centro de Investigación en Matemáticas <https://orcid.org/0000-0002-5431-5927>

Gara Miranda

Universidad de La Laguna <https://orcid.org/0000-0002-2181-0067>

Research Article

Keywords: Linear Ordering Problem, Diversity-Aware Optimizers, Memetic Algorithms, Combinatorial Optimization

Posted Date: January 12th, 2022

DOI: <https://doi.org/10.21203/rs.3.rs-1248038/v1>

License:  This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

A Diversity-Aware Memetic Algorithm for the Linear Ordering Problem: Improving Best-Known Solutions for Standard Benchmarks

Lázaro Lugo^{1†}, Carlos Segura^{1*†} and Gara Miranda^{2†}

¹Área de Computación, Centro de Investigación en Matemáticas (CIMAT), Callejón Jalisco s/n, Mineral de Valenciana, Guanajuato, 36240, Guanajuato, México.

²Departamento de Ingeniería Informática y de Sistemas, Universidad de La Laguna, Avda. Astrofísico Francisco Sánchez, s/n, San Cristóbal de La Laguna, 38200, Santa Cruz de Tenerife, España.

*Corresponding author(s). E-mail(s): carlos.segura@cimat.mx;

†These authors contributed equally to this work.

Abstract

The Linear Ordering Problem (LOP) is a very popular NP-hard combinatorial optimization problem with many practical applications that may require the use of large instances. The Linear Ordering Library (LOLIB) gathers a set of standard benchmarks widely used in the validation of solvers for the LOP. Among them, the xLOLIB2 collects some of the largest and most challenging instances in current literature. In this work, we present new best-known solutions for each of the 200 complex instances that comprises xLOLIB2. Moreover, the proposal devised in this research is able to achieve all current best-known solutions in the rest of instances of LOLIB and improve them in other 93 cases out of 485, meaning that important advances in terms of quality and robustness are attained. This important advance in the field of the LOP has been possible thanks to the development of a novel Memetic Algorithm (MA) that was designed by taking into account some of the weaknesses of state-of-the-art LOP solvers. One of the keys to success is that the novel proposal allows for a gradual shift from

exploration to exploitation, which is done by taking into account the stopping criterion and elapsed period of execution to alter the internal decisions taken by the optimizer. The novel diversity-aware proposal is called the Memetic Algorithm with Explicit Diversity Management (MA-EDM) and extensive comparisons against state-of-the-art techniques provide insights into the reasons for the superiority of MA-EDM.

Keywords: Linear Ordering Problem, Diversity-Aware Optimizers, Memetic Algorithms, Combinatorial Optimization

1 Introduction

The Linear Ordering Problem (LOP) is a classical combinatorial optimization problem that arose in the field of economics. It has applications in other research areas, such as machine learning and logistics. The problem was initially introduced in 1936 by Leontief [1] and formalized in 1958 by Chenery and Watanabe [2], but it was not until decades later that Garey and Johnson [3] demonstrated that LOP is an NP-hard problem. Although the problem has multiple applications and formulations, it is quite common to define the problem as a matrix triangulation problem.

Given a matrix $M_{n \times n} = (m_{ij})$, the triangulation problem is to determine a simultaneous permutation σ of the rows and columns of M such that the sum of the entries above the main diagonal is maximized (or equivalently, the sum of subdiagonal entries is minimized) [4]. Hence, the LOP can be formulated as finding a permutation σ that maximizes the following equation:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^n m_{\sigma_i \sigma_j} \quad (1)$$

Several exact and approximate solvers have been proposed for this combinatorial optimization problem [5]. The interest in this problem has continued over the years, resulting in research compilations such as [4]. So far, different algorithms have been devised that yield very promising solutions for the LOP instances addressed in [4]. Within these proposals, the Memetic Algorithms (MAs) have proven to be very effective in the search for improved solutions to the LOP [6]. As described in the following section, most trajectory- and population-based state-of-the-art approaches for the LOP rely on restarting procedures or other mechanisms to regain diversity or to abandon oversampled regions, in an effort to avoid stagnation and premature convergence. In addition, they usually include problem-dependent information to exploit the particularities of the problem's structure. The performance of schemes based on recovering diversity has been improved in several large-scale problems by methods that explicitly manage diversity [7], and methods that relate the amount of diversity to the elapsed period of execution and stopping criterion have been particularly successful [8]. However, this last design principle has not

been adopted in current LOP solvers. The hypothesis of this research is that there is still a big room for improvement in the design of LOP solvers for large instances, and thus, the solutions that are used as a reference when validating LOP solvers might be enhanced significantly. Knowing near-optimal values is important to properly measure the capabilities of different solvers. Thus, this research aims to design a novel long-term solver that is able to obtain new best-known solutions for the most challenging instances of the LOP. Given the promising behavior of MAs and the lack of population-based LOP solvers that establish a dependency between the amount of diversity maintained in the population and the elapsed execution period and stopping criterion, a novel MA that adopts this design principle is applied.

The novel diversity-aware memetic algorithm proposed in this research is called the Memetic Algorithm with Explicit Diversity Management (MA-EDM). MA-EDM combines an effective local search and well-tested permutation-based genetic operators with a novel diversity-aware replacement strategy that aims to gradually shift from exploration to exploitation as the evolution progresses. As shown in the experimental validation, quite a large number of new best-known solutions were attained for the most popular instances used in the literature. This is quite an important achievement, especially considering that these benchmarks have been available for decades and that many different kinds of methodologies have been used to tackle them. It is also worth noting that our proposal does not incorporate some of the problem-dependent knowledge about the structure of the problem that is taken into account in the most recent LOP solvers. Thus, this research also shows that by applying proper general metaheuristic design principles, simpler and more effective LOP optimizers can be developed.

In the following section, the state of the art in the LOP area is summarized by introducing a chronological description of the most promising approaches and their main features. Afterwards, our novel diversity-aware memetic algorithm is described in detail. Finally, the experimental analysis is presented and the main conclusions are drawn. The experimental validation takes into account the best-performing metaheuristics proposed in the literature with the aim of showing that the advances are not only due to the fact that long-term executions are carried out. The advantages of MA-EDM are quite remarkable. Experiments to shed some light on the reasons behind the superiority of MA-EDM are also included.

2 Literature review

A wide variety of approaches to deal with the LOP have been proposed [5], including exact methods like branch-and-bound or cutting planes, specific heuristics and multiple metaheuristic strategies. In recent years, metaheuristics have gained more attention, especially for dealing with the most recent benchmarks, which include quite complex and large instances [6, 9]. The most relevant metaheuristics developed for the LOP are described briefly below.

In 1999, Laguna et al. [10] proposed a Tabu Search (TS) approach which incorporates strategies for search intensification and diversification. Two different definitions of basic neighborhoods and path relinking are introduced for search intensification, while the diversification relies on ad-hoc short-term and long-term components. The short-term component is focused on modifying items that have been changed infrequently. Meanwhile, the long-term component is used to perform restarts by generating solutions that are distant from the best solutions found so far. This approach improved — at the time — the state-of-the-art heuristics [11]

As an alternative, Campos et al. [12] proposed a Scatter Search scheme that takes into account diversity in two of its components. First, in order to generate initial diverse solutions, GRASP constructions were designed with a greedy function that combines attractiveness and diversity. Second, the reference set selects half of its members by quality and half of its members by maximizing a measure of diversity. Regarding the improvement method, it is also based on the two neighborhoods proposed in [10], while an ad-hoc combination method based on votes is used.

Later, Schiavinotto and Stützle [9] performed a search-space analysis of the available LOP instances. Depending on certain characteristics of the matrices, such as variability, number of zeros or symmetry, they realized that those based on real data are very different from random ones. Moreover, the LOLIB instances [13] were too small to pose a real challenge to state-of-the-art metaheuristic approaches, and even to exact algorithms, so an additional set of large, random, real-life like instances —denoted as eXtended LOLIB (xLOLIB)— was generated. The most complex instances present a low correlation between fitness and distance to optimal solutions, which suggests that exploring distant regions is important to performance. As a result, an MA that includes an Iterated Local Search (ILS) for intensification and an adaptive restart strategy for diversification was proposed. For the local search, the neighborhoods were defined through the interchange or insert moves, and the swap operation of the current permutation. In addition to these neighborhoods, the sorting-reversing procedure introduced by Chanas and Kobylanski [11] was also applied. Since the performance of an MA may depend strongly on the crossover operator, four different operators were tested. The most promising results were obtained with Cycle Crossover (CX) and Order Crossover (OB). The experimental study showed that the MA resulted in a much more robust performance; in particular, the restarting mechanism was key to the success.

Trajectory-based heuristics were further advanced by developing variants of Variable Neighborhood Search (VNS) [14]. Several variants that include different intensification and diversification approaches were proposed. Experimental analysis shows that the performance of VNS improves when combined with short-term memory TS and diversification schemes. While proper solutions were attained in short-term executions, the solutions were not as promising as those output by MAs.

In 2012, Martí, Reinelt and Duarte [4] published an essential survey in the field of the LOP. Their work offers a comprehensive study of the main existing heuristic and metaheuristic approaches for the LOP at the time. For the heuristic methods, the authors studied some classical construction algorithms and some improvement algorithms based on inserts —moves or exchanges— or on local search. The experimental study also included some multi-start methods [11], as well as all the metaheuristics already introduced here: Tabu Search [10], GRASP [12], Scatter Search [12], Genetic Algorithms [9], MA [9], ILS [9], and VNS [14]. The experimental study also included a Simulated Annealing based on the noising algorithm by Charon and Hudry [15, 16]. For their analysis, the authors also compiled all the existing —and widely used in related literature— LOP benchmarks to release a new version of the LOLIB library. Their statistical analysis demonstrated that MA and ILS [9] constitute the most promising approaches.

Subsequently, the potential of ILS was explored further. Two general frameworks, based on the ILS and the Great Deluge Algorithm respectively, were proposed. The main novelty relies on exploring the neighborhood in a more efficient way [17, 18] with the aim of increasing the amount of iterations that might be evolved.

An alternative to reduce the computational cost of local search-based approaches was proposed by Ceberio, Mendiburu, and Lozano [6]. They conducted a theoretical analysis of neighborhoods that allowed them to identify specific positions where the indexes might generate local optima solutions. This way, a restricted insert neighborhood is considered in conjunction with the best performing algorithms [4], MA and ILS [9], to generate the MA_r and ILS_r methods. The comparisons of MA_r and ILS_r —at the level of solution quality but especially in terms of execution times— demonstrated the superiority of these approaches. In order to go further in their testing, the authors generated an extra set of more complex instances, known as xLOLIB2, that were also integrated in LOLIB. xLOLIB2 is currently the set of largest instances that is most discussed in the related literature.

In order to further improve the exploration behavior of ILS and MA, the Hybrid Exploration Algorithm (HEA) was devised, which is a population-based metaheuristics based on ILS [19]. Similarly to the proposal put forth in this paper, HEA moves the balance from exploration to exploitation during the execution. This is done through a perturbation with a dynamic strength. Note that differently to our proposal, this adaptation does not take into account the stopping criterion or elapsed period. Additionally, a post-optimizer based on exactly solving submatrices of the instance by dealing with the LOP as a binary linear programming problem is applied. New best-known solutions were set to 77 out of the 78 instances from the xLOLIB [9].

Finally, with the aim of exploiting some structural properties of the LOP, Santucci and Ceberio [20] designed a new approach, denoted by CD-RVNS, which involves the following components: a VNS algorithm based on restricted neighborhoods, a randomized heuristic construction procedure that builds-up

an LOP solution precedence by precedence, and a destruction procedure that removes precedences from an LOP solution with the aim of producing a new starting point for the construction procedure. VNS considers two different neighborhoods, *insert* and *interchange*, by switching between them when the search is trapped in a local optimum. During the construction and destruction procedures, an LOP solution is represented as a set of pairwise precedences, thus differing from the classical linear representation of permutations, which is indeed used in the VNS phase. Dealing with both representations requires the implementation of conversion methods between representations. Although representing a linear ordering as a set of precedences requires more memory than encoding a linear permutation, such a representation allows constructive heuristics to smoothly build up the solutions, precedence by precedence, thus avoiding drastic changes in the objective value.

In summary, it is important to remark that most proposals call attention to the importance of diversification with the aim of avoiding stagnation. However, recent design principles based on relating the diversity to the stopping criterion and elapsed period have never been applied in LOP solvers. Additionally, most recent strategies have focused on employing specific features of the structure of the LOP and/or hybridizing metaheuristics with exact solvers to advance the results further, meaning that the complexity of solvers has increased drastically, with algorithms containing several thousand lines of code. Finally, trajectory-based approaches have generally been more successful in short-term executions, whereas in the long term, population-based approaches are usually more convenient. While MA reports most of the current best-known solutions, the trajectory-based ILSr and CD-RVNS solvers, currently offer some of the best-known solutions for several instances, so when validating new solvers, it is important to consider both kinds of metaheuristics.

3 Memetic Algorithm with Explicit Diversity Management: MA-EDM

Knowing near-optimal solutions for large instances is important for the analysis of the performance of LOP solvers. Given the hypothesis previously stated, the research is focused on developing a LOP solver that, in the long-term, is able to find new best-known values for LOP instances of high complexity. Diversity-aware strategies that gradually shift from exploration to exploitation as the evolution progresses have excelled in combinatorial problems [8]. Thus, in order to attain our aim, a novel diversity-aware LOP solver, the Memetic Algorithm with Explicit Diversity Management (MA-EDM) is proposed. MA-EDM was designed by considering some genetic operators that have already been proven effective for the LOP. Algorithm 1 shows the pseudocode of our proposal. It is a first-generation MA, so a static improvement individual strategy is applied. Each component is described below in detail.

Our approach starts by initializing a population with N individuals randomly (line 1). As stated previously, in most effective LOP solvers, solutions

Algorithm 1 Memetic Algorithm with Explicit Diversity Management

Require: *LOP instance*, N (size of population), crossover operator (CX or OB), *Stopping criterion (time)*

- 1: **Initialization:** Generate an initial population P_0 with N individuals. Assign $i = 0$.
 - 2: **Local Search:** Perform a local search for every individual in P_0 .
 - 3: **Diversity Initialization:** Calculate the initial desired minimum distance (D_0) as the mean distance among individuals in P_0 .
 - 4: **while** the execution time has not finished **do**
 - 5: **Mating Selection:** Perform binary tournament selection on P_i in order to fill the mating pool with N parents.
 - 6: **Variation:** Apply the crossover in the mating pool to create the set O_i with N offspring.
 - 7: **Local Search:** Perform a local search for each individual in O_i .
 - 8: **Survivor Selection:** Apply the replacement technique (BNP) to create P_{i+1} by considering P_i and O_i as input.
 - 9: $i = i + 1$
 - 10: **end while**
 - 11: **Return** best evaluated permutation.
-

directly encode the permutation, so this was the representation we selected. The initialization considers a uniform random generator, meaning that each permutation is equiprobable. Then, each solution is improved by applying a first-improvement stochastic hill-climber that considers the insert neighborhood (line 2) [10]. Note that this is one of the most successful neighborhood definitions, and neighbors are generated by simply moving a number of the given permutation to an alternative position while shifting all the numbers in the intermediate positions. Naive implementations of this local search evaluate the whole neighborhood in $O(N^3)$. However, since the best insertion position of a number can be calculated in $O(N)$, neighbors are visited more systematically to reduce the total cost to $O(N^2)$ [17]. Note that similar improvement strategies are applied in most state-of-the-art MAs [18, 20]. It is also important to note that the stochastic behavior of the hill-climber is maintained because at each iteration, the order in which the numbers to move are considered is random. Once a number is selected, if its best insertion move improves the current solution, it is accepted. This is repeated until no neighbor improves the current solution. Note that the local search with the complexity reduction technique allows achieving local optima quite efficiently, even for the largest instances considered in our experimental validation. For example, in instances with $n = 1000$, the time to reach a local optimum, when starting from a randomly created solution, is less than one second. This means that it can be directly integrated into MAs without the need to consider alternative stopping criteria.

An important feature of our approach is that it considers diversity explicitly. This is done with the replacement strategy, which works by setting a minimum desired distance that is updated during the run. The details are given later, but similarly to [21], the distances that appear in the initial population are used to set the initial desired distance (D_0) (line 3). Note that this has the advantage of adapting the execution automatically to the given instance [21], in the sense that usually, for the most complex instances where a higher diversity is required to reach high-quality solutions, the improved initial population tends to exhibit larger distances. D_0 is calculated as the mean distance among all the individuals in the initial population. In order to calculate D_0 , a metric or a distance-like function must be selected. In this case, the permutation deviation distance (D_{dev}) [22] is employed. The dissimilarity between two permutations x and y with p elements is defined as follows:

$$D_{dev}(x, y) = \sum_{i=1}^p \|i - pos(x[i], y)\|$$

where $pos(v, perm[])$ is the index of the value v in the permutation $perm$.

The *permutation deviation distance* was selected because the position of a number establishes the amount of values that are summed from the associated column. Thus, summing the absolute distance between the positions for each number is meaningful. Note that this metric has a broader range than other typical metrics applied to permutation-based representations, such as the exact match distance, which provided important advantages for the Job Shop Scheduling problem [21]. Moreover, it can be calculated in $O(N)$, meaning that the impact on performance is low, which also justifies its selection. Any of the metrics described in [22] could also be used, and some of them might even yield better results. However, given the advantages of the permutation deviation distance mentioned previously, and that this work considers a large amount of complex instances that require computationally expensive experiments, the analyses of alternative metrics is left for future work.

As in standard MAs, MA-EDM evolves a set of generations until a given stopping criterion is reached (line 4-10). In order to provide fair comparisons, and since very different types of proposals, including trajectory-based and population-based approaches, are considered in our validation, the stopping criterion was set by time. At each generation, a set of N parents is selected using binary tournaments (line 5). Then, the variation phase is applied (line 6). In this case, the variation consists only of a crossover operator, and the mutation is omitted. The logic behind this decision is that in many cases, MAs achieve proper solutions even if mutation is not used [23]. The local search performs modifications that allow new alleles to appear, so the increased complexity of including an additional operator is not always justified [24]. This is especially the case when applying diversity-aware strategies [21], because one of the other roles of mutation is to avoid the premature loss of diversity, which in our case is preserved explicitly in the replacement phase. Regarding the crossover operators, two of the most effective are the Cycle Crossover

(CX) [25] and the Order Crossover (OB) [26]. A clear winner is not identified in the related literature [9], so both of them are considered in this research. The CX operator is based on the principle of inheriting values in its exact positions. Thus, each number in the offspring permutations shares its position with one of the two parents. First, both offspring inherit the elements that share the same position in both parents, by just copying them in their original position. Then, the remaining elements are used to establish cycles that fulfill the following property: consecutive elements in the cycles share the same position in the parents. Note that with this property, the cycles can be identified unequivocally. For each number, one of the parents is selected randomly and the elements belonging to the associated cycle are copied in the same positions as in the selected parent. The OB operator is concerned with preserving the partial order among elements. First, one of the parents is cloned. Then, half of the positions are selected randomly, and they are sorted according to the order appearing in the second parent [26]. Note that in order to generate the second offspring, the roles of the parents are interchanged. In order to improve the generated offspring, they are subjected to the same improvement strategy that was applied to the initial population (line 7).

Finally, the survivor selection strategy is applied to select the members of the next population using the Best Non-Penalized (BNP) survivor selection strategy (line 8). The main design principle behind BNP is similar to the one discussed in [8] and relies on introducing a dependency between the selection of survivors and the ratio between the elapsed execution period and the stopping criterion. Specifically, the aim is to promote a larger degree of exploration in the initial phases of the execution, and then to gradually shift towards exploitation as the end of the optimization process nears. Note that BNP was successfully applied in the optimization of the Job Shop Scheduling Problem [21]. However, to our knowledge, it has never been applied in the Linear Ordering Problem.

Algorithm 2 shows the pseudocode of BNP. It receives as its input the current population and offspring, and it outputs the population of the next generation. BNP is based on promoting both quality and diversity, and the second aim is attained by avoiding the selection of individuals that are too close to each other. The definition of too close is dynamic. Specifically, a threshold distance (D) that is used to identify too-close individuals is calculated with the formula given in line 1. In this formula, $T_{current}$ is the number of elapsed seconds from the start of the optimization process, and T_{total} is the stopping criterion (also in seconds). Thus, the initial distance (D_0) previously discussed is reduced linearly, so that at the end of the optimization process, D is set to zero. This means that, as the evolution progresses, the promotion of diversity is reduced and, consequently, more importance is given to the quality. This transformation is performed gradually, resulting in a steady shift from exploration to intensification. BNP manages three sets of individuals. The eligible set ($S_{Eligible}$) contains individuals that have not yet been selected to survive and that are not too close (distance lower than D) to any of the solutions already selected to survive. The penalized set ($S_{Penalized}$) contains individuals

Algorithm 2 Best Non-Penalized (BNP) Survivor Selection Strategy

Require: Population $P = \{p^1, \dots, p^N\}$, Offspring $O = \{o^1, \dots, o^N\}$

- 1: $D = D_0 - \frac{T_{Current}}{T_{Total}} \times D_0$
- 2: $S_{Eligible} = P \cup O$
- 3: $S_{Penalized} = \emptyset$
- 4: $S_{NewPopulation} = \emptyset$
- 5: **while** $|S_{NewPopulation}| < N$ **do**
- 6: **for** $individual \in S_{Eligible}$ **do**
- 7: $individual.dci = DCI(individual, S_{NewPopulation})$
- 8: **if** $individual.dci < D$ **then**
- 9: $S_{Eligible} = S_{Eligible} \setminus \{individual\}$
- 10: $S_{Penalized} = S_{Penalized} \cup \{individual\}$
- 11: **end if**
- 12: **end for**
- 13: **if** $S_{Eligible} \neq \emptyset$ **then**
- 14: $new_selected = \operatorname{argmax}_{x \in S_{Eligible}} x.obj$ // Ties are broken randomly
- 15: **else**
- 16: **for** $individual \in S_{Penalized}$ **do**
- 17: $individual.dci = DCI(individual, S_{NewPopulation})$
- 18: **end for**
- 19: $new_selected = \operatorname{argmax}_{x \in S_{Penalized}} x.dci$ // Ties are broken randomly
- 20: **end if**
- 21: $S_{NewPopulation} = S_{NewPopulation} \cup \{new_selected\}$
- 22: **end while**
- 23: **return** $S_{NewPopulation}$

that have not yet been selected to survive and whose distance to a survivor is lower than D . Finally, the new population set ($S_{NewPopulation}$) contains the survivors. Initially, $S_{Eligible}$ is filled with the union of the current population and offspring (line 2), whereas $S_{Penalized}$ and $S_{NewPopulation}$ are empty (lines 3 and 4). Finally, the process iteratively selects survivors until $S_{NewPopulation}$ is filled with N individuals (line 5 to 22). At each iteration, the distance condition is checked for each individual in $S_{Eligible}$ (lines 6 and 7), and those that are too close to an already selected survivor are moved to $S_{Penalized}$ (line 8 to 11). Note that in the pseudocode, DCI stands for *Distance to Closest Individual*, and is defined as follows: $DCI(x, P) = \min\{distance(x, y) \mid y \in P\}$, where $distance(x, y)$ refers to the deviation permutation distance already discussed. In those iterations where $S_{Eligible}$ is not empty, i.e. there are sufficiently distant individuals, the best among them (the one that maximizes the objective function) is selected to survive (lines 13-14 and 21). The opposite case represents situations where the diversity is not as large as desired. Thus, the penalized solution with the longest distance to the closest survivor is selected (lines 16-21).

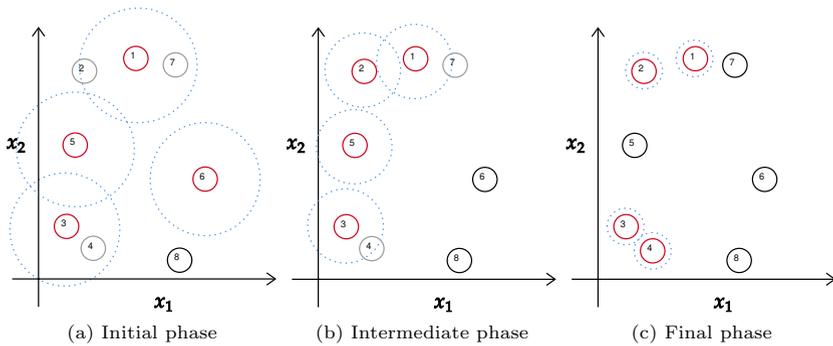


Fig. 1: Illustration of the BNP replacement process for a continuous optimization problem with two variables

In order to further illustrate the replacement operator, Figure 1 shows the behavior of BNP in three different scenarios: initial, intermediate and final phases of the run. Particularly, in each case, it considers that its input is a set of eight solutions (union of previous population and offspring) from which four solutions must be picked up as survivors. Each subfigure represents the status at the end of the BNP process. Note that the threshold distance to perform the penalties is reduced over time, meaning that each of the previous scenarios consider different threshold distances. In order to facilitate the illustration, this figure considers that BNP is applied to a continuous problem with two variables (x_1 and x_2). Each small circle with a number represents a solution and the value inside the circle represents its relative quality, meaning the solution tagged with a 1 is the best one, whereas the one tagged with a 8 is the worst-one. Solutions with a red circle are the survivors ($S_{NewPopulation}$), the ones with a gray circle are the penalized ones ($S_{Penalized}$), whereas the ones with a black circle are not penalized nor selected ($S_{Eligible}$). Also note that the blue circle surrounding each survivor represents the region of the search space that is penalized because of such a selection. Note that, at each iteration of BNP, the best non-penalized solution is selected until four or them are marked in red color. Due to the different threshold sizes, the survivors marked in subfigure (a) are more focused on diversity than on quality, whereas subfigure (c) represents a case with a higher selection pressure. In this particular example, it selects the best solutions even if it implies an important reduction of diversity. One of the keys of BNP is precisely to perform a smooth reduction of the threshold distance, with the aim of shifting gradually from exploration to exploitation.

4 Experimental Study

This section is devoted to validate the advances attained by MA-EDM. One of the main aims of this research is to obtain new best-known solutions for the most complex sets of instances, so that they can be used as new reference

Set	Instances	Size(#Instances)	Optimum Known
IO	50	44(31)	50
		50(4)	
		56(11)	
		60(3)	
		79(1)	
MB	30	100(5)	30
		150(10)	
		200(10)	
		250(5)	
SGB	25	75(25)	25
RandB	90	40(20)	71
		44(50)	
RandA1	100	50(20)	-
		100(25)	
		150(25)	
		200(25)	
RandA2	75	500(25)	25
		100(25)	
		150(25)	
Spec	37	200(25)	30
		11-36(8)	
xLOLIB	78	50-452(29)	-
		150(39)	
xLOLIB2	200	250(39)	-
		300(50)	
		500(50)	
		750(50)	
		1000(50)	

Table 1: Summary information of the sets of instances that comprise LOLIB.

solutions. However, it is also important to show that such solutions are not attainable by state-of-the-art solvers when similar computational resources are granted. Thus, the set of experiments aims to provide new best-known solutions, compare MA-EDM against state-of-the-art solvers and shed some light on the reasons behind the superior performance of MA-EDM.

In the LOP field, several sets of instances with different dimensions and degrees of complexity have been proposed and integrated in LOLIB. The experimental validation considers nine of the most popular benchmarks sets [4], which are the following: IO [13], MB [27], SGB [28], RandB [4], RandA1 [10], RandA2 [12], Spec [4], xLOLIB [9] and xLOLIB2 [6]. Some of them, especially the four first sets of benchmarks, are relatively easy and are almost systematically solved by current solvers. Thus, they are useful just to check the robustness of new proposals. The rest of the benchmark sets contain some larger and more difficult cases, and even though some of them were proposed more than 30 years ago, they have yet not been solved to optimality. Thus, they are normally used to check new advances in the field. Table 1 summarizes some of the most relevant features of the sets of instances included in LOLIB. Particularly, it shows the number of instances that comprises each set,

the sizes of the instances, and the number of instances where the current best-known solution matches the current best-known upper bound. Note that only in the smallest instances, there is a match between the best-known solutions and the best-known upper bounds. As it is shown below, the best-known solutions are improved in this paper for 293 instances. However, the best-known upper bounds were not attained in any of those instances. In fact, the distances between the upper bounds and the best-known solutions is quite large, so probably there is also room for improvement in terms of the upper bounds.

Regarding state-of-the-art solvers, the validation was conducted by considering both trajectory- and population-based metaheuristics. For the trajectory-based methods, the ILSr [6] and CD-RVNS [20] methods were used. Note that CD-RVNS yields better results than ILSr in most cases. However, there are some specific instances where ILSr provides a highly effective search, so it was maintained in our validation. Regarding the population-based alternatives, MAr [9] is the current clear winner, and similarly to MA-EDM, it is also a Memetic Algorithm, so it was selected to perform the validation. Note that the source codes of all these methods are publicly available. In order to facilitate the replicability of our experiments, our source code is also freely available ¹.

Three kinds of experiments are presented below. First, the tuning step of MA-EDM is described. Second, MA-EDM is compared against the selected state-of-the-art solvers. Finally, some analyses that provide a better understanding of the differences between the dynamics of the populations of MAr and MA-EDM are detailed. Note also that in order to provide sound conclusions, and taking into account the stochastic behavior of the solvers involved, all our executions were repeated 30 times and proper statistical tests were applied. Specifically, the results were compared with the *scmamp library* [29] and Scikit-learn [30], following some of the guidelines described in [31]. All the tests were applied assuming a significance level of 5%.

4.1 Parameterization

The parameterization of MA-EDM is a relatively simple task. The two only decisions that had to be made to perform the experimentation were the selection of the crossover operator and the population size. As mentioned earlier, regarding the crossover operator, it was clear that CX and OB had yielded the best overall results [9], so both of them were considered in the tuning phase. Regarding the population size, it is well known that adequate values usually depend on other components of the optimizer, such as the stopping criterion, in non-trivial ways [32, 33]. Since diversity-aware techniques usually require long executions and they alter the dynamics of the population, using previously reported experiments to select the population size is not too meaningful, so we decided to test the following values: 25, 100, 200, 300, 400, 500 and 600.

¹https://github.com/carlossegurag/LOP_MA-EDM. In addition to the source code, the datasets generated and/or analyzed during the current study are available in this repository in the RawData directory. Moreover, this repository contains the best-known solutions generated for each instance in the BKS.zip file.

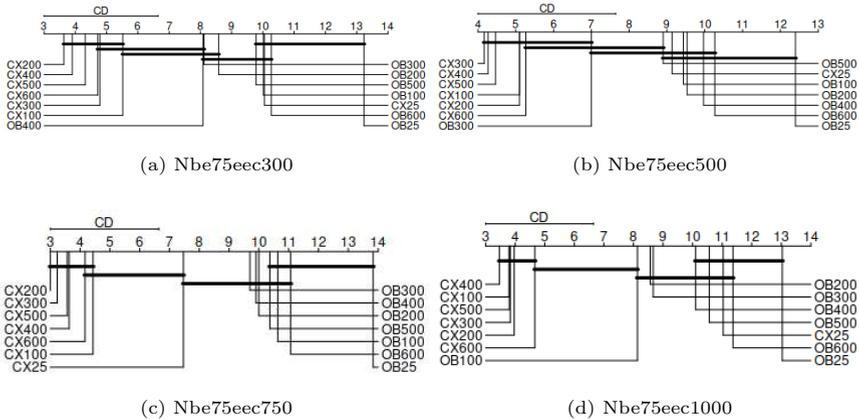


Fig. 2: Critical differences plot for each instance selected, considering the 14 configurations applied in the tuning phase

In order to understand the implications of the parameterization on the results, four instances of the benchmark with the largest instances (xLOLIB2) were considered. Since xLOLIB2 consists of instances with four different sizes, one instance of each size was selected. They were *Nbe75eec300*, *Nbe75eec500*, *Nbe75eec750*, and *Nbe75eec1000*. The fourteen different configurations resulting from combining two crossover operators and seven population sizes were executed 30 times for each instance. The stopping criterion was set to 4 hours.

The results of each instance were analyzed independently. First, the Kruskal Wallis test was applied as an omnibus test. In every case, significant differences appeared so Mann Whitney with Hommel's correction was used to identify the pair-wise differences [34]. Additionally, with the aim of visualizing the results, the Nemenyi test was applied. The critical difference plot of each instance is shown in Figure 2. Note that in this kind of plot, the average rank position of each model is shown and horizontal lines join schemes with no statistically significant differences. Each scheme is denoted with a string formed by the concatenation of the crossover operator and the population size. The advantage of the CX operator is quite clear. In the four instances, at least the six best configurations use the CX operator. Moreover, there was only one instance (*Nbe75eec500*) where the results attained with the best configuration that relies on the OB operator was not statistically different than the ones reported by the best CX operator. Thus, the superiority of the configurations with the CX operator is considerable. As for the population sizes, the conclusions are not as clear. First, an important asset of MA-EDM is that for the configurations with the CX operators, most of the configurations with different population sizes appear linked, meaning that results are not very sensitive to the population size. Only the CX25 configuration is significantly different in some of the instances. Thus, there is quite a large range of admissible

Optimizer	Parameterization
ILSr	$\epsilon = 0.0001$, $i_change = 7$, $n_ni = 750$
MAR	$N = 200$
CD-RVNS	Parameter-less Algorithm
MA-EDM	$N=200$, Crossover Operator=CX

Table 2: Parameterization applied in each state-of-the-art solver and MA-EDM

values. Since CX200 was the best ranked in two instances, and its ranking difference was never larger than half of the critical difference, this was the configuration selected to continue our validation. Note that selecting any of the other configurations in the 100 to 600 range would probably lead to similar conclusions.

Finally, note that while the Nemenyi critical difference plot is quite illustrative, the more advanced statistical tests Kruskal Wallis (as omnibus) and Mann Whitney test with the Hommel's correction (as post-hoc) are more advisable [34]. When comparing CX200 to the remaining configurations with these tests, the same conclusions than the ones obtained with the Nemenyi test could be drawn.

4.2 Validation against the state of the art

One of the most important objectives of our research is to design a proper long-term solver, i.e., an optimizer that is able to profit from long-term executions by avoiding stagnation problems. In order to validate this aim, the three state-of-the-art optimizers selected and the MA-EDM were executed with every instance of the nine benchmark sets by setting the stopping criterion to 4 hours. Table 2 shows the parameterization applied in each optimizer. Note that this is the parameterization recommended by the authors, but in the case of MAR, a parameterization study similar to the one presented above was carried out to set the population size, and the value $N = 200$ was selected. Each execution was repeated 30 times with different seeds.

In order to summarize our results, statistical tests similar to those from the previous experiment were applied. Specifically, the results attained for each instance with the four optimizers tested were compared by using the Kruskal Wallis test as an Omnibus test, followed by the Mann Whitney with the Hommel's correction to detect pair-wise differences. Note that, for every instance, MA-EDM always attained an equal or larger mean and median than the rest of the methods. Table 3 shows, for each benchmark set, the number of instances where the differences between MA-EDM and each state-of-the-art method selected were statistically significant (column $>$) and not statistically significant (column $=$). As we can note, the amount of instances where the results attained by MA-EDM are statistically superior is quite noticeable. Particularly, the superiority appeared in 439, 289 and 420 instances out of the 685 instances, when comparing MA-EDM to ILSr, MAR and CD-RVNS,

	IO		MB		RandA1		RandA2		RandB		SGB		Spec		xLOLIB		xLOLIB2	
	=	>	=	>	=	>	=	>	=	>	=	>	=	>	=	>	=	>
MA-EDM vs ILSr	50	0	29	1	2	98	42	33	90	0	0	25	33	4	0	78	0	200
MA-EDM vs MAr	50	0	30	0	67	33	75	0	90	0	25	0	36	1	23	55	0	200
MA-EDM vs CD-RVNS	50	0	30	0	13	87	51	24	90	0	0	25	31	6	0	78	0	200
Total instances	50		30		100		75		90		25		37		78		200	

Table 3: Summary of statistical tests: column > shows the number of instances where statistically significant differences appeared in favor of MA-EDM, and column = counts the cases with no significant differences

	BKS	ILSr	MAr	CD-RVNS	MA-EDM
Ntiw56n54_300	2654641	2657882	2661516	2650252	26629001
Nt75e11xx_300	145044286	145093367	145297161	144958876	145388484
Nt65f11xx_500	29285209	29284570	29336499	29247869	29365313
Nt75e11xx_500	374756997	374780909	375603945	374381639	375998123
Ntiw56n58_750	20633749	20661491	20704523	20645181	20720453
Nt65b11xx_750	129443488	129523720	129868029	129349693	130011100
Nbe75eec_1000	122183020	122395155	122743873	122255253	122852580
Nt70f11xx_1000	177714930	177754220	178289425	177802295	178387875

Table 4: Mean objective function attained in eight selected instances from xLOLIB2

respectively. In the smallest benchmarks, no much statistically significant differences appeared. However, as the complexity grows, the number of instances with statistically significant differences in favor of MA-EDM clearly increases. Particularly noticeable is the case of xLOLIB2, where significant differences appeared in the whole benchmark set.

Since xLOLIB2 contains the largest instances, eight instances were selected from this benchmark to show the results in more detail. They were two instances from each of the different sizes present in the benchmark set. Table 4 shows the mean objective function attained by each method, as well as the current *Best-Known Solution* (BKS). The improvement provided by MA-EDM is noticeable, and the differences with respect to the BKS are quite significant, which indicates that current solvers are not as effective with these large and complex instances, and it is precisely in the most complex cases where MA-EDM excels. Table 5 shows the amount of new BKS found by each method in the different benchmark sets. Note that when more than one method found the same new BKS, it is counted in all of them. The advantages provided by MA-EDM are quite clear. In fact, there was no instance where MA-EDM was exceeded in terms of the best or mean attained value.

In order to facilitate future validations, the supplementary material attached to this work contains the detailed results for the 685 instances considered over the course of this research. For each instance, the best and mean solution attained by each method is shown. In addition, the BKS is included. MA-EDM was able to set new bounds in 293 cases out of the 685 instances, and it was always able to attain the BKS, meaning that the increase in robustness is huge in comparison to current solvers.

New BKS values per sets of instances					
Instance/Algorithm	ILSr	MAR	CD-RVNS	MA-EDM	# Instances Improved
IO	0	0	0	0	0
MB	0	0	0	0	0
SGB	0	0	0	0	0
Spec	3	3	1	3	3
RandB	0	0	0	0	0
RandA1	2	10	4	38	38
RandA2	1	4	3	4	4
xLOLIB	0	16	5	48	48
xLOLIB2	0	0	0	200	200
Total of new BKS	6	33	13	293	293

Table 5: New BKS found in the experimentation process

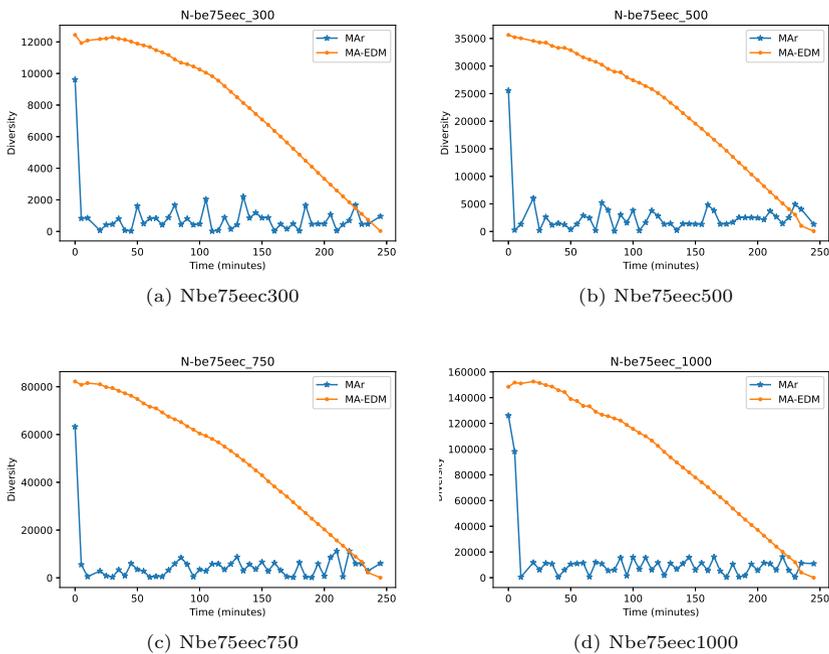


Fig. 3: Evolution of the diversity (mean permutation deviation distance) during the run in four selected instances (mean of 30 executions)

4.3 Analysis of Diversity

One of the most important distinguishing features of MA-EDM is the explicit management of the diversity present in the population. In order to understand the different dynamics of the populations of MA-EDM and MAr, four instances were selected and the intermediate results were saved. Figure 3 shows the trend in the population's diversity during the run. Specifically, the diversity was calculated as the mean of the permutation deviation distances appearing

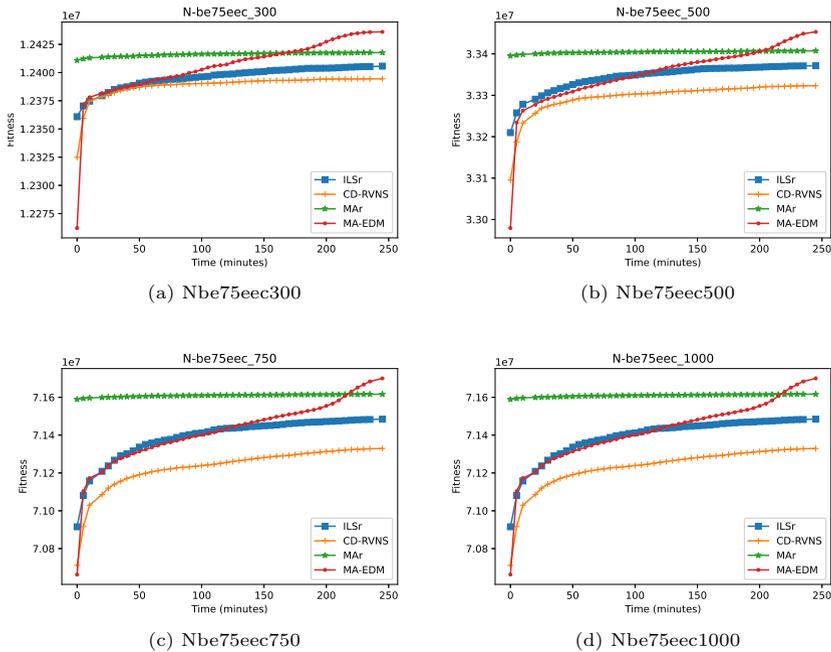


Fig. 4: Evolution of the best objective function attained during the run in four selected instances (mean of 30 executions)

in the population, and the results of 30 independent executions were averaged. The four instances tested showed the same trend. In MA-EDM, the reduction in diversity was almost linear, which is explained by the linear reduction in the desired distance that is considered in the replacement phase. By contrast, MAr reduced its diversity quite quickly, maintaining a low but non-zero diversity. This means that MAr performs the search by placing all the individuals in a relatively narrow region and all the individuals are moved relatively close to one another. In comparison, MA-EDM initially maintains the individuals quite distant from one another, and as the execution progresses, they are gathered in the most promising regions located so far.

In order to complement the above findings, the trend in the best solution evaluated during the search is shown in Figure 4. As in the previous experiment, the mean of 30 executions is shown and, in this case, the trajectory-based methods are also included. First, note that for these long-term executions, population-based approaches are much more effective. Second, it is also clear that MA-EDM exhibits a slow convergence, and specifically, when compared to MAr, it shows a lower solution quality during most of the search. However, as the end of the search nears and the diversity decreases, intensification is promoted and MA-EDM is able to profit from the promising regions identified and yield much better final quality values. Finally, it is also important to note

that these results do not indicate that MA-EDM requires more than 3 hours to improve on the results of MAR. If a more restrictive computational budget were to be applied, MA-EDM would reduce its diversity faster, so even when using shorter times, MA-EDM might also exceed MAR.

5 Conclusions and Future Work

The LOP is a very popular NP-hard combinatorial optimization problem with numerous applications. Metaheuristics have excelled in solving the LOP. However, their performance degrades when considering large LOP instances. Knowing near-optimal solutions is important to properly measure the performance of solvers. This paper represents an effort to improve further the best-known solutions for standard benchmarks used in the validation of LOP solvers. Currently, most state-of-the-art population-based metaheuristics designed for the LOP rely on simple schemes based on partial restarts to avoid premature convergence. However, recent design trends in the field of MAs show that for difficult combinatorial optimization problems, important advances can be achieved when more elaborate diversity management mechanisms are included; specifically, those schemes that alter their internal components to relate the amount of diversity maintained to the ratio between the elapsed execution period and the stopping criterion, have yielded significant improvements. In order to design a LOP solver that is able to improve further the new best-known solutions, a novel MA that includes the mentioned design principle is devised. This proposal is called the Memetic Algorithm with Explicit Diversity Management (MA-EDM). MA-EDM incorporates a novel survivor selection strategy that is based on avoiding individuals that are too close to each other, which is done dynamically and considers the permutation deviation distance.

The experimental validation clearly confirms the advantages provided by this design principle for the LOP. New best-known solutions were attained in 293 well-known instances, and the comparison against the state of the art shows quite remarkable improvements. Moreover, the advantages provided by MA-EDM were confirmed with proper statistical tests, and additional analyses were conducted to understand the reasons behind the important advances in MA-EDM in comparison to alternative MAs. The differences between the amount of diversity maintained in MA-EDM and other MAs is noticeable, which explains the important improvements contributed by MA-EDM. Thus, this research confirms the importance of proper diversity management in LOP solvers, and establishes a significant amount of new best-known solutions that should be taken into account in future validations of LOP solvers.

One of the most important drawbacks of our approach is that high-quality solutions are only attained near the end of the optimization process. Introducing some alternative mechanisms that combine the intensification and exploration phases might improve the anytime behavior of MA-EDM [35], which could allow its application in alternative conditions, where not just the

quality of the final solution is taken into account. Another topic that was not considered in this research was an analysis of the different ways of calculating distances between solutions. Considering alternative metrics such as the exact match distance and the reversal distance seems quite promising. Finally, the supplementary material associated with this research shows that as the dimensionality of the instances increases, the distance between the best and mean objective function attained by MA-EDM grows. Since the neighborhood considered in this paper exhibits quadratic growth, dealing with matrices with several thousand rows and columns might drastically limit the number of generations evolved, which could deteriorate the quality considerably. In order to develop more robust and scalable approaches, techniques that have been devised specifically to improve scalability might be incorporated. Specifically, applying dimensionality reduction techniques and parallelization might allow the attainment of additional new best-known solutions.

Acknowledgments

Authors acknowledge the financial support from CONACyT through the “Ciencia Básica” project no. 285599 and the support from “Laboratorio de Supercómputo del Bajío” through the project 300832 from CONACyT.

References

- [1] Leontief, W.W.: Quantitative Input and Output Relations in the Economic Systems of the United States. *The Review of Economics and Statistics* **18**(3), 105–125 (1936)
- [2] Chenery, H.B., Watanabe, T.: International comparisons of the structure of production. *Econometrica* **26**(4), 487–521 (1958)
- [3] Garey, M.R., Johnson, D.S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., Philadelphia, USA (1979)
- [4] Martí, R., Reinelt, G., Duarte, A.: A benchmark library and a comparison of heuristic methods for the linear ordering problem. *Computational Optimization and Applications* **51**, 1297–1317 (2012)
- [5] Martí, R., Reinelt, G.: *The Linear Ordering Problem - Exact and Heuristic Methods in Combinatorial Optimization*, p. 169. Springer, Springer Heidelberg Dordrecht London New York (2011)
- [6] Ceberio, J., Mendiburu, A., Lozano, J.A.: The linear ordering problem revisited. *European Journal of Operational Research* **241**(3), 686–696 (2015)

- [7] Črepinšek, M., Liu, S.-H., Mernik, M.: Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys* **45**(3), 35–13533 (2013)
- [8] Segura, C., Hernández-Aguirre, A., Luna, F., Alba, E.: Improving diversity in evolutionary algorithms: New best solutions for frequency assignment. *IEEE Transactions on Evolutionary Computation* **21**(4), 539–553 (2016)
- [9] Schiavinotto, T., Stützle, T.: The linear ordering problem: Instances, search space analysis and algorithms. *Journal of Mathematical Modelling and Algorithms* **3**(4), 367–402 (2004)
- [10] Laguna, M., Martí, R., Campos, V.: Intensification and diversification with elite tabu search solutions for the linear ordering problem. *Computers & Operations Research* **26**(12), 1217–1230 (1999)
- [11] Chanas, S., Kobylański, P.: A new heuristic algorithm solving the linear ordering problem. *Computational Optimization and Applications* **6**, 191–205 (1996)
- [12] Campos, V., Glover, F., Laguna, M., Martí, R.: An Experimental Evaluation of a Scatter Search for the Linear Ordering Problem. *Journal of Global Optimization* **21**, 397–414 (2001)
- [13] Grötschel, M., Jünger, M., Reinelt, G.: A cutting plane algorithm for the linear ordering problem. *Operations Research* **32**(6), 1195–1220 (1984)
- [14] García, C.G., Pérez-Brito, D., Campos, V., Martí, R.: Variable neighborhood search for the linear ordering problem. *Computers & Operations Research* **33**(12), 3549–3565 (2006)
- [15] Charon, I., Hudry, O.: A survey on the linear ordering problem for weighted or unweighted tournaments. *4OR* **5**, 5–60 (2007)
- [16] Charon, I., Hudry, O.: An updated survey on the linear ordering problem for weighted or unweighted tournaments. *Annals of Operations Research* **175**, 107–158 (2010)
- [17] Sakuraba, C.S., Yagiura, M.: Efficient local search algorithms for the linear ordering problem. *Int. Trans. Oper. Res.* **17**, 711–737 (2010)
- [18] Sakuraba, C.S., Ronconi, D.P., Birgin, E.G., Yagiura, M.: Metaheuristics for large-scale instances of the linear ordering problem. *Expert Systems with Applications* **42**(9), 4432–4442 (2015)
- [19] Garcia, E., Ceberio, J., Lozano, J.A.: Hybrid heuristics for the linear

- ordering problem. In: 2019 IEEE Congress on Evolutionary Computation (CEC), pp. 1431–1438 (2019)
- [20] Santucci, V., Ceberio, J.: Using pairwise precedences for solving the linear ordering problem. *Applied Soft Computing Journal* **87**, 105998 (2020)
- [21] Hernández Constantino, O., Segura, C.: A parallel memetic algorithm with explicit management of diversity for the job shop scheduling problem. *Applied Intelligence*, 1–13 (2021)
- [22] Sevaux, M., Sörensen, K., *et al.*: Permutation distance measures for memetic algorithms with population management. In: Proceedings of 6th Metaheuristics International Conference (MIC'05), pp. 832–838 (2005)
- [23] Neri, F., Cotta, C., Moscato, P.: *Handbook of Memetic Algorithms*. Springer, Berlin, Heidelberg (2011)
- [24] Tsai, H.-K., Yang, J.-M., Tsai, Y.-F., Kao, C.-Y.: An evolutionary algorithm for large traveling salesman problems. *IEEE Transactions on Systems, Man, and Cybernetics* **34**(4), 1718–1729 (2004)
- [25] Merz, P., Freisleben, B.: Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE transactions on evolutionary computation* **4**(4), 337–352 (2000)
- [26] Davis, L.: *Handbook of Genetic Algorithms*. Van Nostrand Reinhold Company, New York, USA (1991)
- [27] Mitchell, J.E., Borchers, B.: Solving linear ordering problems with a combined interior point/simplex cutting plane algorithm. In: Frenk, H., Roos, K., Terlaky, T., Zhang, S. (eds.) *High Performance Optimization*, pp. 349–366. Springer, Boston, MA (2000)
- [28] Knuth, D.: *The Stanford GraphBase: A Platform for Combinatorial Computing*. ACM Press, New York, USA (1993)
- [29] Calvo, B., Santafé, G.: scmamp: Statistical Comparison of Multiple Algorithms in Multiple Problems. *The R Foundation* **8**(1), 248–256 (2016)
- [30] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Édouard Duchesnay: Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12**(85), 2825–2830 (2011)
- [31] García, S., Fernández, A., Luengo, J., Herrera, F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in

- computational intelligence and data mining: Experimental analysis of power. *Information Sciences* **180**(10), 2044–2064 (2010)
- [32] Eiben, A.E., Smith, J.E.: *Introduction to Evolutionary Computing*. Natural Computing Series. Springer, Berlin, Heidelberg (2015)
- [33] Zhang, Y.-A., Ma, Q., Sakamoto, M., Furutani, H.: Effects of population size on the performance of genetic algorithms and the role of crossover. *Artificial Life and Robotics* **15**(2), 239–243 (2010)
- [34] del Amo, I.G., Pelta, D.A.: SRCS: a technique for comparing multiple algorithms under several factors in Dynamic Optimization Problems. In: *Metaheuristics for Dynamic Optimization*, pp. 61–77. Springer, Berlin, Heidelberg (2013)
- [35] Radulescu, A., López-Ibáñez, M., Stützle, T.: Automatically Improving the Anytime Behaviour of Multiobjective Evolutionary Algorithms. In: Purshouse, R.C., Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J. (eds.) *Evolutionary Multi-Criterion Optimization*, pp. 825–840. Springer, Berlin, Heidelberg (2013)

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- [Segura22MemeticComputingSM.pdf](#)