Kernels of Mallows Models under the Hamming Distance for solving the Quadratic Assignment Problem

Etor Arza[a], Aritz Pérez[a], Ekhiñe Irurozki[a], Josu Ceberio[b]

[a] *BCAM - Basque Center for Applied Mathematics, Spain*

[b] *University of the Basque Country UPV/EHU, Spain*

Abstract

The Quadratic Assignment Problem (QAP) is a well-known permutation-based combinatorial optimization problem with real applications in industrial and logistics environments. Motivated by the challenge that this NP-hard problem represents, it has captured the attention of the optimization community for decades. As a result, a large number of algorithms have been proposed to tackle this problem. Among these, exact methods are only able to solve instances of size $n < 40$. To overcome this limitation, many metaheuristic methods have been applied to the QAP.

In this work, we follow this direction by approaching the QAP through Estimation of Distribution Algorithms (EDAs). Particularly, a non-parametric distance-based exponential probabilistic model is used. Based on the analysis of the characteristics of the QAP, and previous work in the area, we introduce Kernels of Mallows Model under the Hamming distance to the context of EDAs. Conducted experiments point out that the performance of the proposed algorithm in the QAP is superior to (i) the classical EDAs adapted to deal with the QAP, and also (ii) to the specific EDAs proposed in the literature to deal with permutation problems.

1. Introduction

The Quadratic Assignment Problem (QAP) [41] is a well-known combinatorial optimization problem. Along with other problems, such as the traveling salesman problem, the linear ordering problem and the flowshop scheduling problem, it belongs to the family of permutation-based (a permutation is a bijection of the set $\{1, ..., n\}$ onto itself) problems [15]. The QAP has been applied in many different environments over the years, to name but a few notable examples, selecting optimal hospital layouts [33], optimally placing components on circuit boards [57], assigning gates at airports [32] or optimizing data transmission [51].

Sahni and Gonzalez [58] proved that the QAP is an NP-hard optimization problem, and as such, no polynomial-time exact algorithm can solve this problem unless P=NP. In this sense, until recently,

Email addresses: `earza@bcamath.org` (Etor Arza), `aperez@bcamath.org` (Aritz Pérez), `eirurozki@bcamath.org` (Ekhiñe Irurozki), `josu.ceberio@ehu.eus` (Josu Ceberio)

Algorithm 1: Estimation of Distribution Algorithm [15]

```
Parameters:
$\underline{P_s}$: The size of the population used by the EDA.
$\underline{\text{M}}$: The size of the set of selected solutions.
$\underline{\text{S}}$: The number of new solutions generated at each iteration.

1 $D_0 \leftarrow$ initialize population of size $P_s$ and evaluate the population
2 for t=1,2,... until stopping criterion is met do
3     $D^{sel}_{t-1}$ select $M \leq N$ from $D_{t-1}$ according to a selection criterion
4     $p_t(x) = p(x|D^{sol}_{t-1}) \leftarrow$ estimate a probability distribution from $D^{sol}_{t-1}$
5     $D^S_t \leftarrow$ sample S individuals from $p_t(x)$ and evaluate the new individuals
6     $D_t \leftarrow$ create a new population of size $P_s$ from $D_{t-1}$ and $D^S_t$
7 end
```

only a few instances of size up to 36 were solved using exact solution algorithms. In fact, and exceptionally, only some instances of size 64 and 128 have been solved by using a combination of three strategies: reformulation to a suitable Mixed-Integer Linear Programming, exploiting the sparsity and symmetry of some particular instances, and a Branch and Bound algorithm (B&B) [29]. These strategies, however, require the instance to be highly symmetric, and in general, this cannot be guaranteed. A quick literature review shows that the vast majority of exact methods for the QAP are based on the B&B algorithm [10]. In order to overcome the high computation cost required by these algorithms, Astreicher et al. [1] proposed a grid computing implementation of B&B. Under this technique, this algorithm can be distributed over the internet, forming a computational grid, with the advantage of bringing down the costs and increasing the availability of parallel computation power.

Unfortunately, despite the previous improvements, in the general case, it is still computationally unfeasible to use exact algorithms for medium and large size instances ($n > 40$). In response to this drawback, the community of combinatorial optimization has proposed a variety of metaheuristic algorithms to tackle the QAP. A few of these proposed methods include Genetic Algorithms [27, 62], Tabu Search [59, 37], Simulated Annealing [50], Ant Colony Optimization [31], Memetic Algorithms [49] and Particle Swarm Optimization Algorithms [46].

Currently, three of the best performing approaches for the QAP are Cooperative Parallel Tabu Search (CPTS) [38], a Memetic Search algorithm (MS) [8] and the Parallel Hyper-Heuristic on the Grid (PHHG) [26]. In particular, CPTS is based on the successful robust tabu search implementation [61]. This tabu search implementation is simpler and is executed in parallel with a shared pool of solutions, aiming to promote both the quality and the diversity of the solutions available to each tabu search execution. Memetic search algorithms combine population based algorithms with local search procedures. The MS implementation [8] uses a uniform crossover operator, a breakout local search procedure (a local search procedure with perturbation mechanics to overcome local optima) [7], a fitness based replacement strategy, and an adaptive mutation strategy. Lastly, PHHG executes different metaheuristic algorithms in parallel, and based on the performance of those algorithms, repeatedly executes the most successful metaheuristics. Even though these three methods are very different from each other, they all share a property: they are highly complex hybrid procedures.

Not limited to the previous approaches, Estimation of Distribution Algorithms (EDAs) [43] have also been used to solve the QAP. Zhang et al. presented an hybrid approach of guided local search and a first marginal based EDA for the QAP [66]. Pradeepmon et al. [56] applied a hybrid EDA to the keyboard layout problem, which is modeled as a QAP problem. Previous EDA references use hybridization techniques to improve the performance of the algorithms, nevertheless, the probability models used are simplistic and are not suited to the characteristics of the QAP. In this paper, we investigate probability models specially suited to deal with the QAP, and introduce a new EDA proposal based on these models.

An Estimation of Distribution Algorithm (EDA) [43] is a population-based evolutionary algorithm (see Algorithm 1 for the general pseudo-code of EDAs). Starting with an initial population (line 1), a subset of the best solutions is selected (line 2). Subsequently, a probability model is learned based on these selected permutations (line 4). Next, new solutions are sampled from the probability model, and their objective value is computed (line 5). Finally, the new solutions are combined with the selected solutions to create a new population (line 6). This process is repeated until a stopping criterion is met, such as exceeding a certain time constraint or a maximum number of iterations.

As reported frequently in the literature, the behavior of an EDA depends highly on the probability model used in the learn-sample cycle, as this is the core component of the algorithm. When considering permutation problems, EDAs can be classified into three categories according to the probability domain of the model used [16]. In the first group, we have EDAs that were designed to solve problems on the combinatorial domain. Specifically, EDAs that were designed for the set $\{(k_1, ..., k_n) \in \mathbb{N}^n : k_i \leq n \ \forall i\}$ (denoted as $[n]^n$ in this work) can be adapted for the permutation space. This adaptation is based on the idea that the solution space of the QAP (the set of every permutation of size n, $\mathcal{S}^n$) is a subset of $[n]^n$. Therefore, given a set of solutions from $\mathcal{S}^n$, a probability model can be learned in $[n]^n$. Consequently, in order to obtain solutions that are in $\mathcal{S}^n$, the sampling procedure must be adapted to guarantee that the new samples are in $\mathcal{S}^n$.

In the second group, we have EDAs that were originally designed to deal with continuous domain problems. Next, in order to deal with permutations, these EDAs use a mapping $\gamma : \mathbb{R}^n \longrightarrow \mathcal{S}^n$, such that, given a real vector $v \in \mathbb{R}^n$, $\gamma(v)$ denotes the order of the items in v. EDAs for continuous problems transform each permutation σ_i in the population into a real vector v_i, making sure $\gamma(v_i) = \sigma_i$ is satisfied for all the individuals in the population. Then, a probability model is learned on $\mathbb{R}^n$ from these real vectors, and new real vectors are sampled from this probability model. Finally, γ is applied to all the new real vectors, to obtain the new solutions. One of the major drawbacks of these models, as stated by Bosman et al. [9], are the overheads introduced by the sorting of the real vectors, which can be costly in certain situations. Another limitation of the models in this group comes from the large redundancy introduced by the codification, since a permutation can be mapped by infinite real vectors [16].

Finally, in the third group, we have the EDAs that use probability models that define a probability distribution on $\mathcal{S}^n$. Among these, we find probability models based on order statistics, such as the Plackett-Luce [55] and Bradley-Terry [34] models, or those that rely on distance-based distributions on $\mathcal{S}^n$ such as the Mallows Model (MM) [48] and the generalized Mallows Model (GMM) [30]. For these EDAs, the solution space of the problem is also the space onto which the probability distribution is defined, making them a more natural choice. The MM is an exponential distribution that can be seen as analogous to the normal distribution over the group $\mathcal{S}^n$. Recently, it has been shown that the distance-metric under which an MM is defined influences the performance of the EDA [20]. In this sense, most of the MMs presented in the literature are based on the Cayley, Kendall's-τ and Ulam distances. For example, Ceberio et al. have applied many variants of MM based EDAs to different permutation problems, including Cayley based Kernels of MM [22] and Generalized MM [19] on the QAP. In addition, a MM variant for the bi-and tri-objective FSP was proposed by Zangari et al. [65]. In fact, Ceberio et al. obtained state-of-the-art results in the FSP by hybridizing a MM EDA with a variable neighborhood search [17]. This illustrates the importance of developing efficient and adequate probability models.

In addition to the three metrics mentioned above, there are other distance-metrics on $\mathcal{S}^n$ that have not previously been considered in EDAs, such as the Hamming distance-metric [35, 36]. The Hamming distance between two permutations counts the number of point-wise disagreements and is a natural choice for measuring the distance between assignments or matchings.

In this paper, extending the work in [2], we take a step forward in the development of EDAs specific for assignment type permutation problems by using probabilistic models based on the Hamming distance. Specifically, the proposed probabilistic model is a Mallows model-based kernel density that uses the Hamming distance. The goal of this paper is not to present a state-of-the-art algorithm. Instead, a methodological contribution is made to the design of probabilistic models of EDAs for permutations. Particularly, we show why the probability model used in the introduced EDA is suitable for the QAP, one of the most popular assignment type permutation problems. In addition, by studying the properties of the QAP, and based on the experimentation results, we claim that the Hamming-based kernel density probability model proposed in this paper is suited to be used in EDAs for solving assignment problems in the solution space of permutations of a certain size[1].

Another relevant feature of the MM is that it is a unimodal model, and is centered at a given central permutation. The unimodality and symmetry properties imposed by the MM can be too restrictive in certain contexts, not allowing multimodal scenarios to be accurately modelled [45]. However, the MM can be suitable as a building block in more complex models. An alternative that breaks these strong assumptions is the kernel density estimate using Mallows kernels (KMMs). Instead of having a central permutation, KMMs spread the probability mass by using a non-parametric averaging of MMs centered at each solution. This allows the distribution to model probability distributions more accurately over the space of permutations when the strong assumptions of the MM are not fulfilled by the set of solutions.

Taking advantage of this flexibility, the EDA approach presented in this manuscript implements a KMM under the Hamming distance. For the sake of analyzing the performance of the proposed algorithm, we conduct three experiments. First, we compare the proposed approach to other Hamming-based MM approaches. Then, we see how it compares to other EDAs that use probability models specific to $\mathcal{S}^n$. Finally, we show that Hamming KMM EDA is better than other classical EDAs. Specifically, conducted experiments show that the proposed approach is better than other EDAs in the literature in terms of lower Average Relative Deviation Percentage (ARDP). Moreover, the use of both Kernels and Hamming seems to be necessary for the best possible performance.

[1]In order to take a step forward in the design of EDAs and avoid misinterpretations, in this paper, we will only consider EDAs in their base form (no hybridization).

The rest of the paper is organized as follows: in the following section, we briefly explain the QAP and the adequacy of the Hamming distance for this problem. Next, in Section 3, we introduce the kernels of Mallows Models over the Hamming distance. Then, in Section 4, we detail the proposed algorithm. Afterwards, in Section 5 we present the experimentation, and Section 6 concludes the article.

2. The Quadratic Assignment Problem and the Hamming distance

The Quadratic Assignment Problem (QAP) is the problem of optimally allocating n facilities at n locations in order to minimize a cost function related to the flow and the distance between every pair of facilities and pair of locations. In the QAP, an instance is defined by two matrices D, $H \in \underset{n\times n}{\mathcal{M}}(\mathbb{R}^+)$, where $D_{i,j}$ is the distance between locations i and j, and $H_{l,k}$ is the flow between facilities l and k. The search space of the QAP is the set of every permutation σ of size n, $\mathcal{S}^n$. In this sense, in the QAP, the aim is to find the permutation $\sigma \in \mathcal{S}^n$ that describes the optimal assignment of facilities into locations, where $\sigma(i) = j$ denotes that the j^{th} facility is assigned to the i^{th} location.

When Koopmans et al. [41] introduced the QAP, they presented it as a maximization problem. Given two cost matrices D, $H \in \underset{n\times n}{\mathcal{M}}$ and a profit matrix $A \in \underset{n\times n}{\mathcal{M}}$, the QAP was formulated as shown in Equation (1).

$$\max_{\sigma\in\mathcal{S}^n} \sum_{i=1}^{n} A_{\sigma(i),\sigma(j)} - \sum_{i=1}^{n}\sum_{j=1}^{n} D_{i,j} H_{\sigma(i),\sigma(j)} \tag{1}$$

Later on, the lineal term $\sum_{i=1}^{n} A_{\sigma(i),\sigma(j)}$ was dropped, since the complexity of the problem lies within the quadratic term $\sum_{j=1}^{n} D_{i,j} H_{\sigma(i),\sigma(j)}$, [47].

$$\min_{\sigma\in\mathcal{S}^n} \sum_{i=1}^{n}\sum_{j=1}^{n} D_{i,j} H_{\sigma(i),\sigma(j)} \tag{2}$$

There is an equivalent formulation that is more popular among exact methods. With this notation, the QAP is formulated as an integer problem:

$$\min \sum_{i,j=1}^{n}\sum_{k,p=1}^{n} D_{i,j} H_{k,p} x_{i,j} x_{k,p} \tag{3}$$

restricted to

$$\sum_{j=1}^{n} x_{i,j} = \sum_{i=1}^{n} x_{i,j} = 1, \quad \forall i,j \in \{1,2,...,n\}$$

$$x_{i,j} \in \{0,1\}, \quad \forall i,j \in \{1,2,...,n\}$$

The parameter term $D_{i,j}H_{k,p}$ can be rewritten as $C_{i,j,p,q} = D_{i,j}H_{k,p}$. By doing so, a generalization of the QAP was proposed [44]. In this generalization, the parameters $C_{i,j,p,q}$ are not a product of two values. Instead, each $C_{i,j,p,q}$ is considered as an independent parameter of a problem instance.

Other more complex variants have been proposed [47], such as the multiobjective QAP [40]. However, in this paper, we only focus on the currently most popular permutation based QAP variant, see Equation (2). We argue that this version of the QAP is the most widely studied version of the QAP.

2.1. Distance-metrics

The MM relies on the definition of the distance for permutations, and three have been primarily considered in the framework of EDAs: Cayley, Kendall's-τ and Ulam [20, 17, 15]. The Cayley distance measures the minimum number of swaps needed to transform a permutation into another one. The Kendall's-τ distance measures the number of differently arranged pairs of items between two permutations. Finally, the Ulam distance between permutations σ and π is equal to the size of the permutations, n, minus the length of the longest increasing subsequence in $\sigma\pi^{-1}$. In addition to the previous distance-metrics, the Hamming metric also been reported for the case of permutations. The Hamming distance between two permutations, σ and π, counts the number of point-wise disagreements they have.

As mentioned in the introduction, the distance employed in the MM critically conditions the performance of the EDA when solving a given problem. Previous work on this topic [20, 39] demonstrated that it is crucial to choose operators (distances, neighborhoods, mutations,...) that better fit the characteristics of the problem. We carried out an experiment in order to analyze the correlation between the distance at which two permutations are and the number of components that differ in both permutations, where component refers to each additive term $D_{i,j}H_{\sigma(i),\sigma(j)}$ for $i, j \in [n]$ in Equation (2). Intuitively, it is preferable when a distance-metric structures solutions in the way that close solutions differ in few components and far away solutions differ greatly. In this way, two solutions that are similar in terms of distance will likely be similar in terms of objective function components. The experiment consists of the following: For each of the considered four metrics, we choose two permutations at distance k from each other. Then, we measured the maximum number of different components they can have on a problem of size $n = 20$, independently of the instance. Finally, we repeated this process several times with different permutations, at the same distance k, in order to obtain the median and the interquartile range of the values. The results are depicted in Figure 1.

As can be observed, the Hamming distance-metric shows the best results among the considered distance-metrics. On the one hand, it presents the least number of different components at each distance k. On the other hand, the number of different components is the same for all the permutations at distance k (unlike the rest of the metrics).

The experiment above demonstrated that, taking the definition of the QAP into account, Hamming is the best option. However, considering specific instances of the problem, for many different reasons, the previous conclusion might not hold. For instance, some components could be identical, producing no change; or the change of some components could be compensated by others. For that reason, in a new experiment, we will analyze the objective function transition for each of the metrics on specific instances of the problem. To that end, starting from a random permutation, we

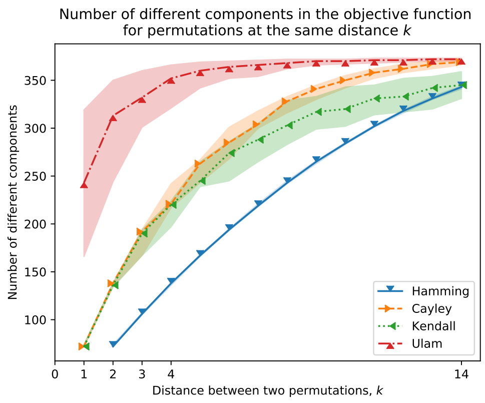


Figure 1: The median, 25% and 75% percentiles of the maximum number of different components that two solutions at a given distance can have in an instance of size $n = 20$. The Hamming distance has the lowest number of different components among the four distance-metrics studied. In addition, the Hamming distance has the most consistent (almost constant) number of different components at a given distance, followed by the Cayley and Kendall distances. The Ulam distance is the worst distance in terms of number of different components.

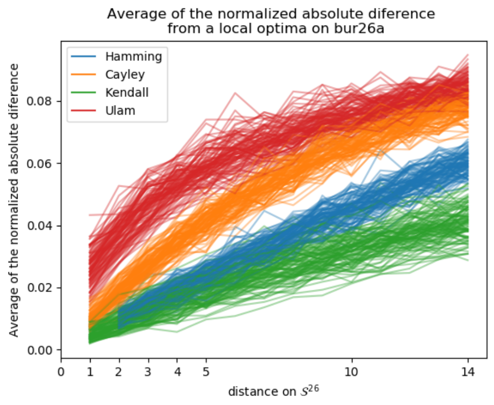
run a local search algorithm[2] to find a local optimum. Then, for each of the four distance-metrics, (Hamming, Cayley, Kendall's-τ and Ulam), the average normalized difference in the objective value with respect to the local optimum is computed for $\forall k \in [14]$. Specifically, defining σ_0 as the local optimum, for each of the metrics, we approximate the difference $\psi_k^{-1} \sum_{\sigma \in \mathcal{S}^n \ | \ d(\sigma_0, \sigma)=k} \frac{\text{abs}(f(\sigma_0)-f(\sigma))}{f(\sigma_0)}$ with the Monte Carlo sampling method using 50 repetitions, where ψ_k is the number of permutations at distance k. In addition to the average, the variance is also computed. The results are deployed[3] in Figure 2.

We observed that Hamming shows a smoother objective value transition than Cayley and Ulam, and, most of the times, than Kendall's-τ (in 4 out of 6 instances) also. Considering the results of these two experiments, shown in Figure 1 and Figure 2, it seems that the Hamming distance is the best choice among the studied metrics for the QAP.

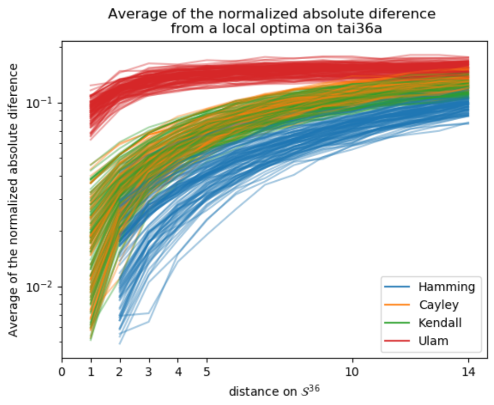
3. Distance-Based Probability Models

Probability models for permutations assign a probability value to each of the permutations of n items. For the sake of applicability and computational efficiency, these probability models are defined by a restricted number of parameters. The Mallows Model (MM) [48] is an exponential probabilistic model defined over $\mathcal{S}^n$. The MM is described by two parameters: the concentration parameter $\theta \in \mathbb{R}^+$, and the location parameter, $\sigma_0 \in \mathcal{S}^n$ [36]. The location parameter, also known as the central permutation, is the mode of the distribution. For the rest of the permutations, their

[2]We used the best-first local search procedure, based on the exchange neighborhood.
[3]For this experiment, 6 instances from the QAPLIB are considered. Figure 2 shows the results obtained for two of them. The full results of the experimentation as well as the source code of the proposed approach are available for the interested reader at `https://github.com/EtorArza/SupplementaryKMMHamming`



(a) bur26a



(b) tai36a

Figure 2: The results of the objective function value transition experiment for two of the instances studied. The instance *bur26a* (a) has special properties on the distance matrix D, which we believe makes the Kendall distance have a smoother objective value transition. The instance *tai36a* (b), on the other hand, does not have these properties, and thus, the Hamming distance produces a smoother objective value transition in this case.

probability decreases exponentially with respect to their distance from the central permutation. The speed of this exponential decay is controlled by θ. For instance, when $\theta = 0$, the distribution is equivalent to the uniform distribution over $\mathcal{S}^n$. Contrarily, when $\theta \to \infty$, $p(\sigma_0) = 1$. Formally, the probability mass function is given as follows:

$$p(\sigma) = p(\sigma|\sigma_0, \theta) = \frac{e^{-\theta d(\sigma, \sigma_0)}}{\psi(\theta)} \tag{4}$$

where $d(\cdot, \cdot)$ is a distance-metric on $\mathcal{S}^n$ and $\psi(\theta)$ stands for the normalization constant.

3.1. Factorization and sampling under the Hamming distance

In the following, we extend the presentation of the MM for the case of the Hamming distance describing the factorization of the probability distribution induced by the model and the procedure to sample the solutions [36]. Under this factorization, a simple sampling procedure for the Hamming MM can be defined. In addition, this decomposition allows a better understanding of the dynamics of the proposed EDA. Defining $\mathcal{K} \equiv d(\sigma_0, \sigma)$ as the Hamming distance from the consensus to σ, we can think of $\mathcal{K}$ and σ as random variables defined in $\{0\} \cup [n]$ and $\mathcal{S}^n$ respectively. From this point of view, $\mathcal{K}$ is dependent on σ, or in other words, given σ, $\mathcal{K}$ is known. Considering this, we can decompose $p(\sigma)$ as:

$$p(\sigma) = p(\sigma|\mathcal{K})\ p(\mathcal{K}) \tag{5}$$

where the first term of the factorization, $p(\sigma|\mathcal{K})$, denotes the probability of σ given the distance at which it is from the consensus, and the second term, $p(\mathcal{K})$, defines the probability of $k = d(\sigma_0, \sigma)$.

The conditional probability distribution of the first term of the factorization shown in Equation (5), $p(\sigma|\mathcal{K})$, follows a uniform distribution. This is easy to see, since the MM gives the same probability to all permutations that are at the same distance k from the consensus. Conveniently, in $\mathcal{S}^n$, the number of permutations at Hamming distance k from a given permutation, $S(n,k)$, can be easily computed. This sequence is closely related to the number of derangements of size k. A derangement is a permutation, σ, where every item σ_i is different from its corresponding index i, hence, $\sigma = (\sigma_1, ..., \sigma_k)$ is a derangement of size k iff $\sigma_i \neq i\ ,\ \forall i \in [k]$. For example, the permutation, $\tau = (\tau^1, \tau^2, \tau^3) = (3,2,1)$ is not a derangement, because $\tau^2 = 2$, while $\gamma = (\gamma^1, \gamma^2, \gamma^3) = (3,1,2)$ is a derangement, because $\gamma^i \neq i\ ,\ \forall i \in \{1,2,3\}$.

In order to compute $S(n,k)$, the following formula can be used:

$$S(n,k) = \binom{n}{k} D(k) \tag{6}$$

where $D(k)$ is the number of derangements of size k [35], which is a known sequence [60]. Specifically, the number of derangements $D(k)$ can be recursively computed in $\mathcal{O}(k)$ as follows:

$$D(k) = \begin{cases} 1 & k = 0 \\ 0 & k = 1 \\ (k-1)(D(k-1) + D(k-2)) & k > 1 \end{cases}$$

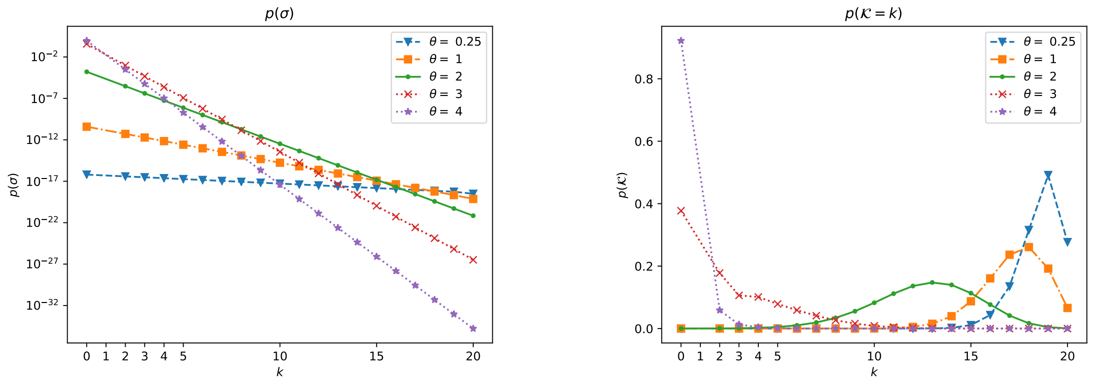
Since we are interested in the first $n+1$ elements of the sequence, we must compute $D(k)$ for $0 \leq k \leq n$, and that requires $\mathcal{O}(n)$ time. Therefore, it is easy to compute the conditional probability $p(\sigma|\mathcal{K} = k) = S(n,k)^{-1}$ for any σ at distance k from the consensus.

Now, we compute $p(\mathcal{K})$, the second term of the decomposition of $p(\sigma)$ in Equation (5). Considering the definition of $p(\mathcal{K} = k) \equiv p(d(\sigma_0, \sigma) = k)$ we obtain:

$$p(\mathcal{K} = k) = \sum_{\sigma | d(\sigma_0, \sigma) = k} p(\sigma) = S(n,k) p(\sigma) = S(n,k) \frac{e^{-\theta k}}{\psi(\theta)} \tag{7}$$

The previous equation can be computed in $\mathcal{O}(n)$ time for $k \in \{0\} \cup [n]$, allowing a simple two-step sampling procedure for the Hamming MM to be defined [36]. First, considering the probabilities $p(\mathcal{K})$, randomly choose k, the distance at which to sample. Secondly, choose a permutation σ at distance k from the consensus σ_0 uniformly at random. A detailed explanation of the sampling procedure is shown later, in Section 4, in Algorithm 2.

The concentration parameter θ controls where the probability of the permutations is concentrated. For a high value of θ, the probability mass is concentrated near the consensus. Similarly, for a low value of θ, the probability mass is concentrated far away from the consensus. This is possible because a low θ defines an almost uniform distribution on $\mathcal{S}^n$, and the number of permutations at Hamming distance k from the consensus increases exponentially with k. Figure 3a shows $p(\sigma)$ described in Equation (4). For high values of θ, $p(\mathcal{K})$ has a higher probability in lower values on k, and vice versa. In Figure 3b, we see that by using different values of θ, the probability mass



(a) $p(\sigma)$ (b) $p(\mathcal{K})$

Figure 3: The representations of $p(\sigma)$ and $p(\mathcal{K})$ on $\mathcal{S}^{20}$ for a Hamming-based MM, considering permutations at Hamming distance $k \in \{0\} \cup [20]$ from the consensus. An instance of size $n = 20$ and different θ values are considered.

of $p(\mathcal{K})$ is concentrated at different distances. This is related with $\mathbb{E}[\mathcal{K}] = \sum_{k\in\{0\}\cup[n]} k\ p(\mathcal{K} = k)$, the expectation of the distance. Given n, the instance size, there exists a bijection that maps the expected distance $\mathbb{E}[\mathcal{K}]$ to the corresponding value of the concentration parameter θ. When $\mathbb{E}[\mathcal{K}]$ is low, the probability of the solutions near the consensus is high, and, consequently, the concentration parameter of the distribution, θ, is high. As we will later see in Section 4.2, by adjusting $\mathbb{E}[\mathcal{K}]$ (and, consequently, its corresponding θ) a simple exploration-exploitation scheme can be defined.

3.2. Extending the Mallows Model

The Mallows Model is a unimodal distribution, and as such, it may be too rigid for multimodal problems, limiting the performance of the EDA in certain situations. As a more flexible alternative, the Generalized Mallows Model was proposed [30] for the Hamming distance [36]. Based on the decomposition property of some distances, the GMM has a unique central parameter, just as the MM, but it also has several concentration parameters, giving the model a higher flexibility.

Introduced in [52], a multimodal alternative to the MM is the Mixtures of Mallows Model (MMM). In this case, the population is considered to be composed of m differently sized clusters. Given the central and concentration parameters for each cluster, σ_i and θ_i, the probability mass distribution is expressed as

$$P(\sigma|\boldsymbol{\sigma}, \boldsymbol{\theta}) = \sum_{i=1}^{m} w_i \frac{e^{-\theta_i d(\sigma,\sigma_i)}}{\psi(\theta_i)}$$

where $\sum_{i=1}^{m} w_i = 1$, $\boldsymbol{w} = \{w_1, ..., w_m\}$ and $w_i > 0$. Usually, $\boldsymbol{\sigma}$, $\boldsymbol{\theta}$ and $\boldsymbol{w}$ are estimated using the Expectation Maximization algorithm [3]. An extension of the MMM that considers several concentration parameters per central permutation is the Mixtures of Generalized Mallows Model (MGMM) [24].

Taking the idea of mixture models to the limit, and by considering each solution in the set as a cluster of equal weight, another even more flexible model can be defined: Kernels of Mallows Model (KMM). Given a set of m permutations, the KMM is the averaging of m MMs centered on these permutations. Therefore, we say that KMM is a combination of several MM. Contrary to the GMM, the KMM is an MM with the same concentration parameter but different central permutations. This model breaks the strong unimodality assumption of the MM and the GMM. Given the set of central permutations $\boldsymbol{\sigma} = \{\sigma_1, \sigma_2, ..., \sigma_m\}$, the mass probability distribution of KMM can be defined as follows:

$$P(\sigma|\boldsymbol{\sigma}, \theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{e^{-\theta d(\sigma,\sigma_i)}}{\psi(\theta)}$$

where $\psi(\theta)$ is the normalization constant.

4. Hamming Kernels of Mallows Model EDA

In this paper, we approach the QAP with an MM-based EDA. Specifically, Kernels of Mallows Model under the Hamming distance are used in the framework of EDAs. To control the convergence of the algorithm, a simple yet effective exploration-exploitation scheme is presented, based on θ, the concentration parameter.

4.1. Learning and sampling

The learning of a model in an EDA usually refers to obtaining the maximum likelihood estimators for the parameters of the selected probabilistic model. Since the proposed EDA is based on Kernels of Mallows Model, a non-parametric model, we do not need to estimate the central permutations. Instead, the selected set of permutations is used as the set of central permutations $\boldsymbol{\sigma} = D_{t-1}^{sel}$. In addition, the concentration parameter θ is set by a simple exploration-exploitation, as we extensively explain in the next section.

Once the model is defined, we need to know how to sample solutions from it. In this case, the sampling procedure is based on the distances sampling algorithm [35], as shown in Algorithm 2. It is a three-step procedure. First, select a central permutation σ_0 from the selected set of permutations $\boldsymbol{\sigma}$ uniformly at random (line 1). Then, based on the probabilities obtained from Equation (7), choose a distance k at which to sample (lines 2 and 3). Finally, a permutation at Hamming distance k from σ_0 is chosen uniformly at random (line 4).

4.2. Exploration-Exploitation scheme: updating θ

The convergence of the EDA is controlled by a simple exploration-exploitation scheme. The trade-off is balanced by the expectation of the distance, $\mathbb{E}[\mathcal{K}]$, which is transformed into its equivalent θ at run-time. The advantages of using $\mathbb{E}[\mathcal{K}]$ instead of θ are threefold. First, we believe $\mathbb{E}[\mathcal{K}]$ is more intuitive than θ, since its interpretation is much easier. In addition, by using $\mathbb{E}[\mathcal{K}]$, it is easier to take into account the instance size n when increasing $\mathbb{E}[\mathcal{K}]$. Finally, $\mathbb{E}[\mathcal{K}]$ is more correlated with

Algorithm 2: Sampling algorithm of Hamming KMM

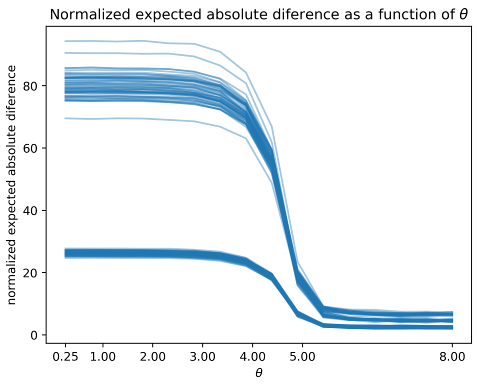
```
   Input:
   $\boldsymbol{\sigma} = \{\sigma_1, ..., \sigma_m\}$: The set of central permutations.
   $\theta$: The concentration parameter.
   Output:
   $\sigma$: The sampled permutation.
1  $\sigma_0 \leftarrow$ choose uniformly at random from $\boldsymbol{\sigma}$
2  compute $\{p(k) \propto S(n,k)e^{-\theta k} \mid k \in [n] \setminus \{1\}\}$
3  $k \leftarrow$ randomly choose $k$ with probability proportional to $p(k)$ (0 is never chosen to avoid
     sampling the consensus.)
4  $\sigma \leftarrow$ choose $k$ items from $\sigma_0$ and derange them (shuffle them uniformly at random, but
     making sure none of these $k$ items remains in its original place)
5  return $\sigma$
```

the transition of the objective function value than θ. Figure 4 shows the evolution of the expected difference in the objective function value and $\mathbb{E}[\mathcal{K}]$ with respect to the concentration parameter θ for an instance of size $n = 125$ (*tai125e01* from the Taixxeyy instances set [28]). It can be seen that the shape of $\mathbb{E}[\mathcal{K}]$ resembles the normalized expected difference of the objective function. In fact, Figure 5 shows that the relationship between $\mathbb{E}[\mathcal{K}]$ and the normalized objective function difference is almost lineal. This means that the transition of the objective value can be more accurately controlled by $\mathbb{E}[\mathcal{K}]$.

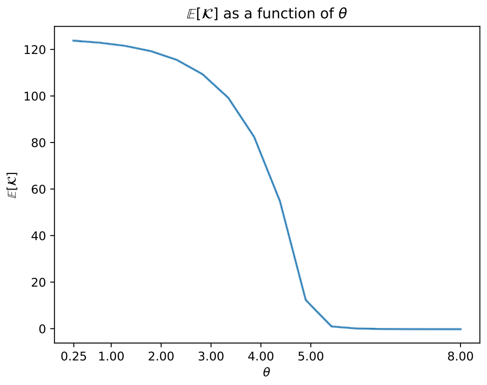
The starting and final values of $\mathbb{E}[\mathcal{K}]_t$[4], $\mathbb{E}[\mathcal{K}]_0$ and $\mathbb{E}[\mathcal{K}]_{tmax}$, respectively, are set before the algorithm is executed. In this sense, $\mathbb{E}[\mathcal{K}]_0$ is set to a high value. A high value of $\mathbb{E}[\mathcal{K}]_t$ favors exploration, because the sampled solutions are expected to be far away from the selected solutions. Therefore, the sampled solutions are going to be very different from the selected solutions, forcing them to visit different and unobserved areas of the solution space. At each iteration, the expectation of the distance $\mathbb{E}[\mathcal{K}]$ is decreased ($\mathbb{E}[\mathcal{K}]_{t+1} < \mathbb{E}[\mathcal{K}]_t$). As the number of iterations increases, the algorithm shifts from an exploration state to an exploitation state. In this exploitation stage, the new solutions will be similar (they will be near each other in the Hamming distance sense) to the known solutions.

An idea to update $\mathbb{E}[\mathcal{K}]_t$ would be to decrease it at a constant rate. However, we found that decreasing $\mathbb{E}[\mathcal{K}]_t$ at an exponential rate produces better results, as we will later discuss in Section 5. The stopping criteria for the algorithm is given in terms of the maximum number of iterations, and the number of solutions evaluated in each iteration is $P_s/2$, where P_s denotes the population size of the EDA. Therefore, at each iteration t, the progress of the algorithm $p \in (0,1)$ is defined as $p = t/t_{max}$. Then, given the intensity parameter $\gamma \in \mathbb{R}^+$, this progress is transformed into an exponential progress with the function $\delta(p) = \frac{\exp(-\gamma p)-1}{\exp(-\gamma)-1}$. Finally, the expectation at iteration t, $\mathbb{E}[\mathcal{K}]_t$, is set to $\mathbb{E}[\mathcal{K}]_t = \mathbb{E}[\mathcal{K}]_{tmax} + \delta(p)(\mathbb{E}[\mathcal{K}]_0 - \mathbb{E}[\mathcal{K}]_{tmax})$. Figure 6 shows $\delta(p)$ for the estimated optimal value of the parameter $\gamma = 5.14$.

[4]$\mathbb{E}[\mathcal{K}]_t$ denotes the expectation of the distance $\mathbb{E}[\mathcal{K}]$ at iteration t.



(a) Normalized expected absolute difference between local optima and solutions sampled using the correspondent θ.



(b) The expectation of d, $\mathbb{E}[\mathcal{K}]$ with respect to θ.

Figure 4: Figure 4a shows the normalized expected difference of the objective function value. This difference is measured between 100 random local optima and solutions sampled using an MM centered on these local optima and concentration parameter θ. Specifically, for each of the considered local optima σ_0, Figure 4a shows $\lim_{s\to\infty} s^{-1}\sum_{i=1}^{s}\frac{\text{abs}(f(\sigma_0)-f(\sigma_i))}{f(\sigma_0)}$ where σ_i is obtained by sampling from an MM centered on σ_0 and using the concentration parameter θ for each $i \in [s]$. The instance tai125e01 was used to obtain this figure. Figure 4b shows the expectation of d, $\mathbb{E}[\mathcal{K}]$ as a function of θ. Observe how the shape of $\mathbb{E}[\mathcal{K}]$ resembles the normalized expected difference of the objective function value.

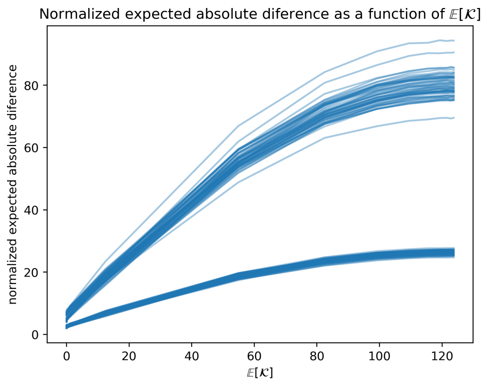


Figure 5: The normalized expected difference of the objective function value as a function of $\mathbb{E}[\mathcal{K}]$.

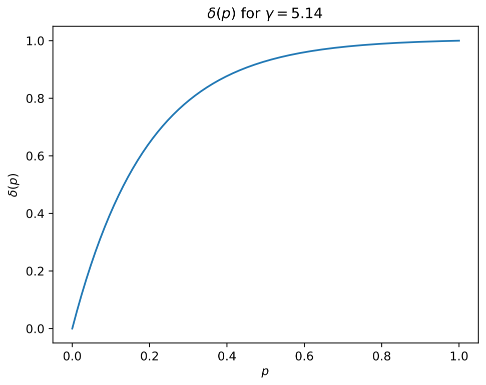


Figure 6: The progress after the exponential function δ is applied.

4.3. Computational complexity and scalability

If n is the instance size and m the considered population size, the time complexity of the sampling stage is $\mathcal{O}(mn)$ [35]. The total cost of the algorithm, without considering the objective function evaluations, is $\mathcal{O}(mn) + \mathcal{O}(m \log(m))$. Considering a constant population size m, the cost of Hamming KMM EDA is dominated by the cost of the evaluations, which is $\mathcal{O}(mn^2)$. The memory complexity of Hamming KMM EDA is $\mathcal{O}(mn)$.

Even though the theoretical computational complexity of Hamming KMM EDA is reasonable (since it is dominated by the cost of the evaluations, $\mathcal{O}(m \cdot n^2)$), a detailed empirical analysis is recommended to study the scalability of the algorithm. To this end, we conducted two experiments on the runtime of the proposed algorithm. In the first experiment, we computed the percentage of time spent evaluating the objective function when using Hamming KMM to optimize *Taixxa* instances of size n, with n ranging from 10 to 100. The stopping criterion was set to $1000 \cdot n^2$ evaluations, and the rest of the parameters remain equal to those considered in the experimental section. Results are depicted in Figure 7. As can be observed, as the instance size increases, the percentage of time spent evaluating the objective function also increases, up to 80% in instances of size 100. This means that the runtime of Hamming KMM is indeed dominated by the evaluation of the objective function, which is a positive feature, since it means that the runtime overhead of the EDA is small in comparison to the cost of evaluating the solutions.

In order to further prove this point, an additional experiment was conducted. Specifically, in this second experiment, we empirically show that the cost of the Hamming KMM EDA algorithm is at most $\mathcal{O}(n^2)$ for a fixed population size of $m = 972$, where n is the problem instance size. With this aim in mind, we run Hamming KMM in instances obtained by cutting *Tai256c*, the largest instance available in the QAPLIB [12]. In this experiment, the stopping criterion was set to 10^6 evaluations, and the rest of the parameters remain equal to those considered in the experimental section. In Figure 8, the proportional time spent on each function evaluation divided by n^2 is shown. Specifically, $\frac{runtime}{E \cdot n^2}$ is shown, where E is the number of evaluations used as stopping criterion ($E = 10^6$), and *runtime* is the time it takes to optimize an instance of size n. Results point out that $\frac{runtime}{E \cdot n^2}$ decreases as n increases. This means that for a fixed stopping criterion in

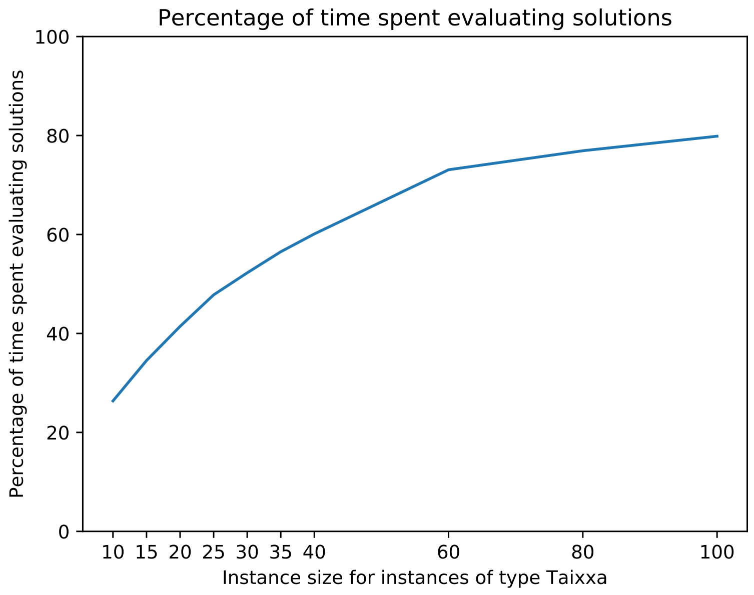


Figure 7: Percentage of time spent on evaluating solutions when optimizing *Taixxa* instances of size n, with $1000n^2$ as stopping criterion.

terms of maximum evaluations and a fixed population size, the actual time complexity of Hamming KMM is at most $\mathcal{O}(n^2)$. In the following, we show how this cost can be reduced even further.

Even though the cost of evaluating a candidate solution is $\mathcal{O}(n^2)$, given two different permutations $\sigma_a, \sigma_b \in \mathcal{S}^n$, if $\sigma_b \in \mathcal{N}(\sigma_a)$ and the objective function value of σ_a is known, then the objective function value of σ_b can be updated in $\mathcal{O}(n)$ time [53]. The proposition below defines the objective function relationship that two solutions at Hamming distance two have.

Proposition. *Suppose* $\exists i_1, i_2 \in [n] | \sigma_a(i) = \sigma_b(i) \quad \forall i \in [n] \setminus \{i_1, i_2\}$. *Then,*

$$\begin{aligned} f(\sigma_b) \;&=\; f(\sigma_a) + \sum_{k \in \{i_1,i_2\}} \sum_{i=0}^{n-1} \left(D_{i,k} H_{\sigma_b(i),\sigma_b(k)} + D_{k,i} H_{\sigma_b(k),\sigma_b(i)}\right) - \\ &\sum_{k \in \{i_1,i_2\}} \sum_{i=0}^{n-1} \left(D_{i,k} H_{\sigma_a(i),\sigma_a(k)} + D_{k,i} H_{\sigma_a(k),\sigma_a(i)}\right) + \sum_{k \in \{i_1,i_2\}} D_{k,k} H_{\sigma_a(k),\sigma_a(k)} - D_{k,k} H_{\sigma_b(k),\sigma_b(k)} \end{aligned}$$

This process can be repeated over and over again to compute the objective function value of permutations at Hamming distance two or more, and if the permutations are close enough, it is more efficient than directly computing $f(\sigma_b)$. It is worth noting that the proposed approach is based on MM kernels and we used the Distances Sampling Algorithm as the sampling procedure [35]. Hence,

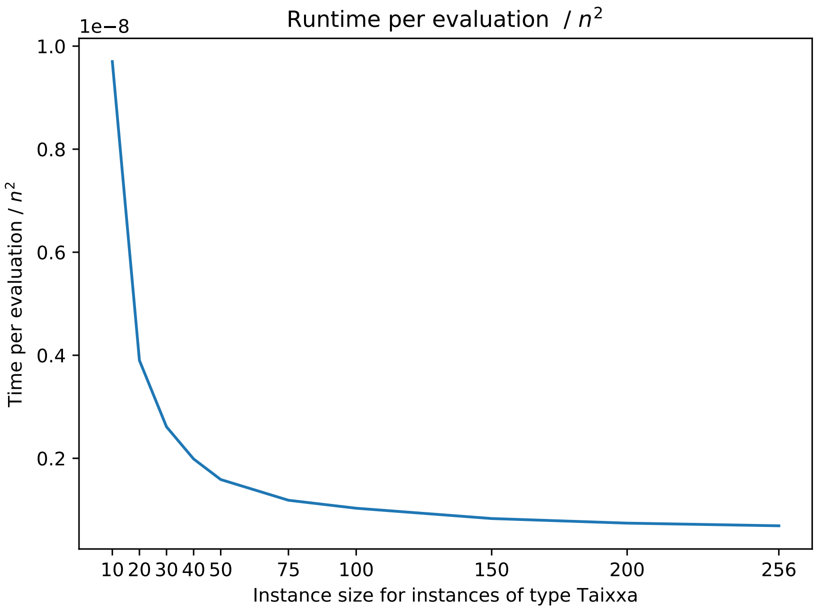


Figure 8: Runtime per evaluation divided by n^2 of Hamming KMM when optimizing *Taixxc* instances of size n, with 10^6 evaluations as stopping criterion.

we sample at a given distance of a known permutation. Therefore, this efficient method to compute the objective function yields a considerable speedup in the EDA, especially in the last iterations of the EDA, where the expected distance from the central permutation to the sampled permutation $\mathbb{E}[\mathcal{K}]$ is small.

5. Experimental study

In order to prove the validity of the proposed method, in this section, we present an exhaustive analysis of the performance of the algorithm.

5.1. General remarks

Some popular instances of the QAP have been employed to evaluate the performance of the proposed approach (Hamming KMM EDA). In particular, 30 instances from the QAPLIB [12] were used.

Before running the experiments, there are a number of parameters that need to be set in the proposed approach. First, we have the starting $\mathbb{E}[\mathcal{K}]_0$ and final $\mathbb{E}[\mathcal{K}]_{max}$ values of $\mathbb{E}[\mathcal{K}]$. $\mathbb{E}[\mathcal{K}]_0$ is set to $n/2$, where n is the instance size, and $\mathbb{E}[\mathcal{K}]_{max}$ is set to 0.25. Setting $\mathbb{E}[\mathcal{K}]_0$ to $n/2$ produces a distribution in which, on average, the sampled permutations have half the items in the same position as the reference permutation σ_0. The chosen $\mathbb{E}[\mathcal{K}]_{max}$ value produces a similar distribution that $\mathbb{E}[\mathcal{K}]_{max} \to 0$ would, but without numerical errors, it is thus the most exploitative state possible. The other two parameters are P_s and γ. The parameter P_s is the population size of the EDA, and γ measures the speed at which $\mathbb{E}[\mathcal{K}]$ is decreased. These two parameters are set using Bayesian optimization [54] with the instance *tai31a* (which is not among the benchmark instances considered). The optimal values found for these parameters are 972 and 5.14 respectively. These parameters are used in all the executions of Hamming KMM EDA.

All the algorithms considered in the experimentation are tested in the set of 30 instances. The stopping criterion is the same for all the considered algorithms and instances: $1000n^2$ evaluations. The experimentation was conducted on a single machine with an octa-core AMD Ryzen 7 1800X Processor, with 8Gb of RAM. Hamming KMM was implemented in C++. The rest of the algorithms were implemented in either Java or C++. In any case, since the number of evaluations is used as the stopping criteria, it is not affected by the hardware nor the programming language, and therefore, is easy to reproduce. As a reference, it takes Hamming KMM about 107 seconds to perform $1000n^2$ evaluations in the largest of the studied instances (tai100a) and 0.1 seconds in the smallest one (tai10a).

For each benchmark instance, the results are recorded as the Average Relative Deviation Percentage, ARDP $= |\frac{f_{best}-f_{av}}{f_{best}}|$, where f_{best} is the best known value and f_{av} is the average of the best objective values obtained in each repetition. For further statistical analysis, Bayesian Performance Analysis [14, 13] (BPA) is carried out to study the uncertainty of the results of each experiment[5] Specifically, Placket-Luce is used as the probability model, defined in $\mathcal{S}^n$, in this case, corresponding to the rankings of the algorithms. BPA considers probability distributions over probability distributions. In our case, assuming a uniform prior distribution, the posterior distribution of the probability of each algorithm being the best performing one (winning) is computed. The goal of this analysis is to determine which algorithm performs the best for the set of considered benchmark instances. The inference with the BPA approach is fairly simple [14]. First the scores of the algorithms are transformed from ARDP to their corresponding rankings on each of the test instances. Then, a sample is produced from the posterior probability distribution of the weights of the Plackett-Luce model (a probability model for rankings). And finally, based on these sampled weights, the credible interval of 90% is computed for each algorithm. This interval means that there is a 90% chance that the actual probability of the algorithm being the highest ranking algorithm (being the winner) lies within the interval.

5.2. Experiment 1: Kernels and the exponential $\mathbb{E}[\mathcal{K}]$

For the sake of measuring the contribution of each of the two main parts that extend a Hamming Mallows Model EDA, (i) the use of kernels and (ii) the use of an exponential increase of $\mathbb{E}[\mathcal{K}]$, we compare the performance of the full model with the simplified variants. The simplified models considered are: KMM with linear increase of $\mathbb{E}[\mathcal{K}]$, MM with exponential increase of $\mathbb{E}[\mathcal{K}]$, both with one missing part; MM with linear increase of $\mathbb{E}[\mathcal{K}]$, missing both parts; and finally, a simple Hamming MM in which the concentration parameter θ is estimated at each iteration. Figure 9 shows all the studied simplified models, ordered by their complexity in terms of the number of free parameters. The average ARDP obtained in all the instances for each of the models is also shown

[5]In the file "comparison_between_BPA_and_NHST.pdf", available in the GitHub repository, we justify the use of Bayesian performance analysis instead of other more traditional hypothesis tests.

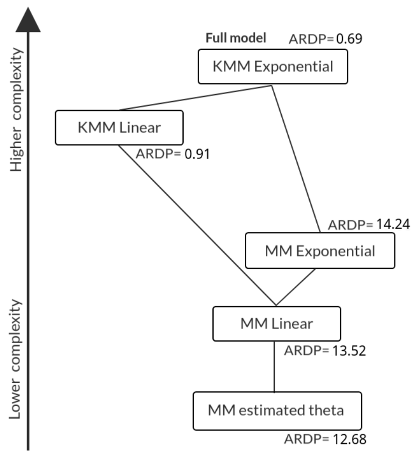


Figure 9: A diagram of the simplified models considered in this paper. The vertical axis is proportional the complexity level of the model in terms of the number of free parameters. The average ARDP obtained in the studied instances is shown for each model.

in this figure.

The same parameters are considered for all the EDAs, thus, the parameters estimated with Bayesian optimization for the full model are used. The ARDP values are recorded in Table 1.

The full model outperforms the rest of the models in 60% of the instances, while the second best model, (KMM with linear increase), only outperforms the rest of the models in 23.3% of the studied instances. It is worth noting that using the kernels part is much more important than the exponential increase of $\mathbb{E}[\mathcal{K}]$, since, as seen in Figure 9, both kernel models have an average ARDP lower than 1%, while the rest of the models have an ARDP over 10%. Additionally, the exponential increase is detrimental for the MM, and it is only a positive addition when we consider it alongside the kernels parts. This experiment indicates that both the kernels and the exponential increase of $\mathbb{E}[\mathcal{K}]$ are key parts of the proposed model.

Figure 10 shows the credible interval of 90% calculated from the samples obtained from the posterior distribution of the probability of each of the algorithms being the best. As can be observed, both kernel-based algorithms (with exponential increase and linear increase) have a higher expected probability of being the best than the rest, around 0.6 and 0.4 respectively.

5.3. Experiment 2: Comparing specific EDAs for permutation problems

In this experiment, we compare Hamming KMM EDA to other specific EDAs for permutation problems[6] considered in the literature. We compare the performance of the proposed approach with respect to other MM-based approaches. For example, the MM has already been applied to permutation problems under the Cayley, Kendall's-τ and Ulam distances [20]. Cayley and Kendall-based KMMs [23] and GMMs have also been studied. Additionally, *mixtures of GMMs* (MGMMs)

[6]Specific EDAs are those that estimate a probability distribution explicitly on $\mathcal{S}_n$.

Table 1: The results of the simplified models when compared with the full model. The best performing algorithm is highlighted in bold.

Instance	KMM Exponential	KMM Linear	MM Exponential	MM Linear	MM estimated θ
bur26a	0.105	**0.100**	1.600	1.469	1.420
bur26b	0.182	**0.165**	1.596	1.534	1.335
bur26c	**0.007**	0.010	2.202	1.970	1.717
bur26d	**0.007**	**0.007**	2.342	2.147	1.930
nug17	0.179	**0.110**	9.919	9.371	8.655
nug18	**0.326**	0.409	10.415	10.332	9.798
nug20	**0.125**	0.175	11.008	10.911	10.381
nug21	0.271	**0.197**	14.475	13.515	12.806
tai10a	**0.000**	**0.000**	3.484	3.207	2.338
tai10b	**0.000**	**0.000**	2.814	2.653	1.524
tai12a	0.140	**0.000**	9.624	8.806	8.280
tai12b	**0.000**	**0.000**	4.924	4.974	4.635
tai15a	**0.179**	0.190	8.252	8.194	7.536
tai15b	**0.007**	**0.007**	0.910	1.020	0.822
tai20a	**0.843**	0.947	12.830	12.379	11.863
tai20b	**0.068**	0.123	9.349	9.089	9.165
tai25a	**1.265**	1.732	13.586	12.941	12.454
tai25b	**0.025**	0.041	30.511	27.680	21.286
tai30a	**1.435**	1.891	12.547	12.258	11.957
tai30b	0.189	**0.075**	34.105	29.592	23.616
tai35a	**1.485**	2.429	13.852	13.121	12.800
tai35b	**0.476**	0.526	30.253	27.398	24.357
tai40a	**1.762**	2.622	13.829	13.451	13.157
tai40b	1.068	**0.299**	37.258	34.402	33.339
tai60a	**2.237**	3.400	13.757	13.524	13.180
tai60b	**0.493**	0.647	33.766	32.900	33.237
tai80a	**2.172**	3.658	12.547	12.361	12.005
tai80b	**2.235**	2.707	33.004	32.617	32.004
tai100a	**2.190**	3.538	11.771	11.619	11.485
tai100b	**1.142**	1.404	30.624	30.156	31.466

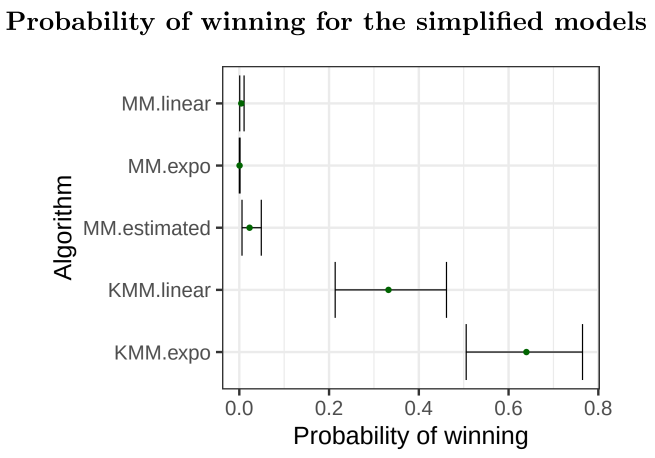


Figure 10: Credible intervals of 90% and expected value of the estimated posterior probability of each algorithm being the winner among those tested.

Table 2: The average ARPD results of Hamming KMM EDA and other EDA approaches specific to S^n. The best performing algorithm is highlighted in bold.

	Hamming	Ulam	Kendall				Cayley				Plackett-
Instance	KMM	MM	MM	KMM	MGMM	GMM	MM	KMM	MGMM	GMM	Luce
		[20]	[20]	[23]	[24]	[18]	[20]	[23]	[24]	[18]	[21]
bur26a	**0.105**	3.716	1.937	2.453	1.955	1.746	0.359	0.211	0.690	0.365	1.730
bur26b	**0.182**	4.052	2.008	2.353	1.998	1.461	0.482	0.305	0.603	0.465	1.680
bur26c	**0.007**	4.386	2.121	2.655	2.054	1.827	0.369	0.172	0.634	0.307	1.843
bur26d	**0.007**	4.756	2.352	2.872	2.233	1.882	0.370	0.168	0.695	0.355	1.890
nug17	**0.179**	9.434	9.174	12.535	9.284	7.771	4.706	2.154	6.680	3.147	7.188
nug18	**0.326**	17.601	8.881	12.746	10.306	8.316	4.689	2.948	6.161	3.684	7.850
nug20	**0.125**	11.391	9.603	11.961	8.541	7.759	5.195	1.844	2.977	3.214	10.895
nug21	**0.271**	13.934	12.252	12.613	12.137	10.738	5.722	2.695	7.859	3.380	13.659
tai10a	**0.000**	15.514	6.464	11.306	8.048	8.395	2.354	1.038	2.244	2.348	2.278
tai10b	**0.000**	25.309	7.025	15.845	7.506	6.327	1.032	1.440	1.475	2.324	2.835
tai12a	**0.140**	9.521	11.905	17.115	12.359	11.496	7.226	5.856	7.415	6.415	6.928
tai12b	**0.000**	6.043	11.716	21.591	13.287	11.565	5.046	3.551	8.264	6.414	6.757
tai15a	**0.179**	7.919	9.225	11.174	9.670	9.128	4.648	3.153	6.497	3.586	5.631
tai15b	**0.007**	0.975	1.279	1.536	1.224	0.997	0.528	0.369	0.749	0.411	0.743
tai20a	**0.843**	12.014	12.050	12.921	11.443	10.942	6.971	2.820	5.412	5.355	11.958
tai20b	**0.068**	7.494	13.348	32.965	12.735	12.788	5.112	5.322	2.345	2.646	6.912
tai25a	**1.265**	12.355	11.856	12.439	11.765	11.159	7.572	4.735	8.397	5.537	11.962
tai25b	**0.025**	16.446	24.200	56.513	30.102	22.254	6.071	5.456	10.529	4.692	20.214
tai30a	**1.435**	15.292	11.263	11.314	10.529	10.184	6.628	3.301	4.629	4.947	11.682
tai30b	**0.189**	49.972	29.100	45.465	27.863	21.636	9.282	7.843	10.025	11.077	22.984
tai35a	**1.485**	12.912	11.879	11.820	11.862	11.221	7.316	4.592	7.621	4.880	12.921
tai35b	**0.476**	20.097	24.583	36.410	29.819	21.667	7.083	5.976	9.532	5.346	25.320
tai40a	**1.762**	13.257	11.651	11.546	11.599	11.004	7.162	3.670	4.904	4.862	13.272
tai40b	**1.068**	28.524	30.422	44.129	33.423	25.576	10.729	8.421	6.970	8.703	33.436
tai60a	**2.237**	13.222	11.367	10.996	11.026	10.234	7.354	3.878	4.666	4.574	13.103
tai60b	**0.493**	34.853	33.144	40.119	33.344	24.317	7.112	5.491	5.897	5.238	32.017
tai80a	**2.172**	12.109	9.964	9.740	9.853	9.465	6.525	3.745	4.111	4.358	12.074
tai80b	**2.235**	32.741	31.416	33.977	30.511	26.626	6.674	5.295	6.053	5.792	32.458
tai100a	**2.190**	11.496	9.469	9.062	9.282	8.711	6.237	3.460	3.617	3.913	11.469
tai100b	**1.142**	31.929	25.849	31.703	29.561	22.020	5.469	4.603	4.888	4.982	31.116

have also been applied to permutation-based problems under the Cayley and Kendall distances [24]. Specifically, in this last article, the MGMM with two clusters was found to outperform the other MGMM approaches, and therefore, we will only consider MGMMs with two clusters. In addition to MM EDAs, we compare the performance of the proposed algorithm to a Plackett-Luce EDA [21].

The ARDP value for all the instances is recorded in Table 2. We kept the parameters proposed by each author in the paper that the algorithm is proposed.

Figure 11 shows the credible interval of 90% of the posterior distribution of the probability of being the best algorithm for the EDAs specific to $\mathcal{S}^n$. Considering credible intervals of 90%, we can say the probability of Hamming KMM being the highest ranked method is higher than 60%. From this analysis, it is clear that Hamming KMM is the algorithm with the highest probability of being the best, followed by Cayley KMM.

To sum up, we observe that KMM EDA obtains a lower ARDP for all the 30 benchmark instances considered. Therefore, the experimentation suggests that using the proposed model is the best option among the specific EDAs on $\mathcal{S}_n$ for the 30 instances considered in this paper.

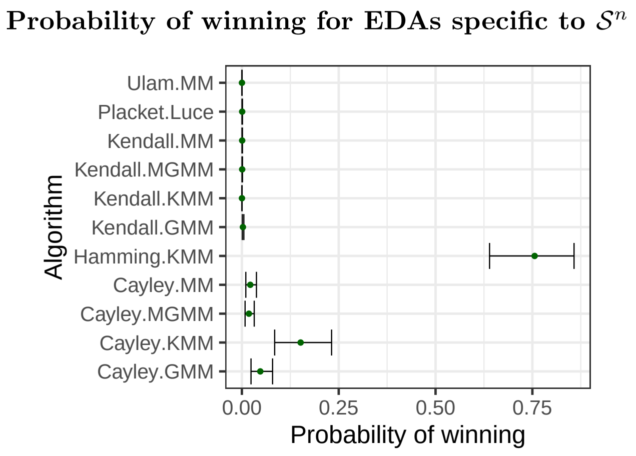


Figure 11: Credible intervals of 90% and expected value of the estimated posterior probability of each algorithm being the winner among those tested.

5.4. Experiment 3: Classical EDAs

Finally, we compare Hamming KMM EDA to other classical EDAs for the QAP in the literature. In the review paper on EDAs in permutation problems [16], the performance of 13 classical EDAs was studied. Using a null-hypothesis statistical testing, the authors found that there were no statistically significant differences among the best performing six methods for the QAP. These six methods are *univariate marginal distribution algorithm* (UMDA) [42], *mutual information maximization for input clustering* (MIMIC) [25], *estimation of Bayesian network algorithm* (EBNA) [6], *edge histogram-based sampling algorithm* (EHBSA) [63] and two variants of *node histogram-based sampling algorithm* (NHBSA) [64], namely NHBSA$_{WT}$ and NHBSA$_{WO}$. In this experiment, we compare Hamming KMM EDA to these other algorithms. Not limited to the previous algorithms, a recent successful EDA, the Random Key EDA [4, 5], was also incorporated to the study.

The parameters proposed by the EDAs review article [16] are used for the methods compared in that article, and we use the parameters proposed by the authors in the RK-EDA [4].

Hamming KMM EDA obtains the best results in terms of a lower ARPD value in 70% of the considered instances. The second most competitive approach is NHBSA$_{WT}$, which outperforms the rest of the methods in 20% of the considered instances. In addition to obtaining the lowest ARPD, Hamming KMM EDA is also the most consistent algorithm. For instance, while NHBSA obtains an ARPD over 5% in one instance, the proposed approach obtains lower than 2.5% ARPD in all instances. However, for the four *bur26x* instances considered, the *node histogram-based sampling algorithm* (NHBSA$_{WT}$) is able to outperform Hamming KMM EDA. These instances have special properties in the distance matrix D. Specifically, adjacent rows and columns are similar to each other. Although Hamming KMM EDA is still the second best approach in these instances, we believe that the Hamming distance is not particularly suited for these instances, as argued in Section 2.1.

Figure 12 shows the credible interval of 90% of the posterior distribution of the probability of being the best algorithm for the EDA not specific to $\mathcal{S}^n$. We can say with high confidence that the

Table 3: The average ARPD results of Hamming KMM EDA and other EDA approaches. The best performing algorithm is highlighted in bold.

Instance	Hamming KMM	UMDA [42]	MIMIC [25]	EBNA [6]	EHBSA$_{wt}$ [63]	NHBSA$_{wt}$ [64]	NHBSA$_{wo}$ [64]	RK-EDA [4, 5]
bur26a	0.105	0.323	0.281	0.311	0.442	**0.094**	0.172	0.535
bur26b	0.182	0.327	0.306	0.387	0.304	**0.095**	0.238	0.475
bur26c	0.007	0.064	0.102	0.116	0.208	**0.000**	0.023	0.356
bur26d	0.007	0.063	0.146	0.073	0.021	**0.000**	0.029	0.213
nug17	**0.179**	2.760	2.200	2.673	1.386	0.202	1.247	2.991
nug18	**0.326**	2.979	3.114	2.663	2.073	0.332	1.917	2.684
nug20	**0.125**	3.070	3.459	2.926	2.023	0.479	1.374	2.907
nug21	0.271	2.022	2.806	1.989	3.199	**0.254**	1.214	3.868
tai10a	**0.000**	2.113	3.295	2.833	1.729	0.043	1.944	5.279
tai10b	**0.000**	0.807	2.282	0.837	**0.000**	**0.000**	0.461	6.617
tai12a	0.140	4.980	5.514	4.690	**0.000**	0.208	4.136	7.181
tai12b	**0.000**	4.100	3.706	3.125	**0.000**	0.055	1.184	10.605
tai15a	**0.179**	2.993	3.634	3.415	3.043	0.665	2.151	4.643
tai15b	0.007	0.250	0.406	0.419	0.373	**0.000**	0.163	8.724
tai20a	**0.843**	4.779	5.226	4.224	4.885	2.280	3.360	7.049
tai20b	**0.068**	3.530	4.450	3.840	1.956	0.270	4.220	4.176
tai25a	**1.265**	4.387	4.700	4.297	6.160	3.630	3.325	6.510
tai25b	**0.025**	2.740	3.462	2.728	1.366	0.099	0.824	9.949
tai30a	**1.435**	3.559	4.643	4.091	6.666	3.896	2.640	6.895
tai30b	**0.189**	6.502	10.143	6.621	1.332	0.765	10.801	16.502
tai35a	**1.485**	4.226	4.976	4.025	7.514	4.919	2.606	7.233
tai35b	**0.476**	4.087	6.355	3.453	2.744	1.162	3.723	7.972
tai40a	**1.762**	4.038	5.246	3.771	7.959	5.292	2.748	7.814
tai40b	**1.068**	5.732	8.221	5.932	4.486	2.418	5.334	9.287
tai60a	**2.237**	4.032	5.001	4.009	7.419	4.243	3.346	7.449
tai60b	**0.493**	1.188	5.309	2.558	8.177	0.836	3.379	7.347
tai80a	**2.172**	3.737	4.621	3.595	7.114	4.415	3.032	7.046
tai80b	**2.235**	4.387	5.491	5.276	12.488	2.340	3.290	8.640
tai100a	**2.190**	3.460	4.227	3.321	6.737	4.519	2.792	6.636
tai100b	**1.142**	2.025	4.672	2.353	11.416	1.214	2.469	5.602

Probability of winning for EDA approaches non-specific to $\mathcal{S}^n$

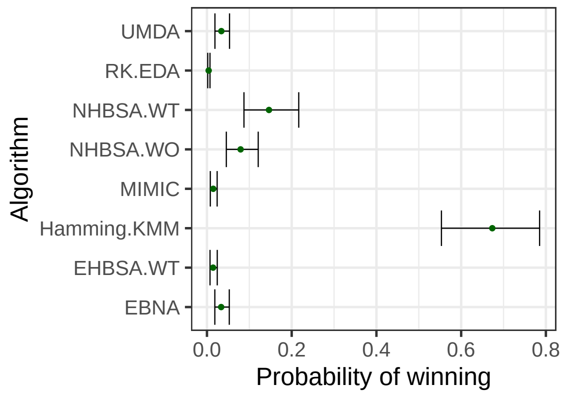


Figure 12: Credible intervals of 90% and expected value of the estimated posterior probability of each algorithm being the winner among those tested.

probability of Hamming KMM being the highest ranked method is above 50%. In contrast, the probability of the next best performing method, $NHBSA_{wt}$, being the best one is lower than 30%. The rest of the methods have a fairly lower performance, with probability below 0.3 of being the highest ranking methods, considering credible intervals of 90%.

Taking into account this analysis, Hamming KMM has a higher chance of being the highest ranked method than the rest of the methods.

6. Conclusion & future work

In this paper, we aimed to take a step forward in the development of EDAs for permutation problems. We argued that the Hamming distance is suitable for the QAP, as it produces the smoothest objective function transitions when compared to other distance-metrics. After analyzing the adequacy of the Hamming distance for the QAP, we proposed an algorithm that implements a Hamming-based Kernels of Mallows Models (KMMs) EDA. In order to analyze the performance of the proposed approach, we compare it to other non-hybrid EDAs presented in the literature. The conducted experimentation showed that, for the QAP, (i) Hamming KMM EDA performs better than other classical EDAs, (ii) it also performs better than other MM approaches in the literature, and (iii) the use of Kernels on a Hamming-based MM is the key to the successful performance of the algorithm. Specifically, Hamming KMM EDA is able to outperform the rest of the methods in 56.7% of the studied instances. Not only that, but Hamming KMM EDA is also more stable than the competitors in terms of maximum ARDP, with an ARDP value of less than 2.5% in the most difficult instance for this algorithm.

The incorporation of Hamming-based KMMs to the EDA framework in a competitive manner opens new research directions worth considering. For instance, this method could potentially be applied to other permutation problems, and even in non-permutation based combinatorial problems, if the solution space of the problem can be encoded by vectors. Because the Hamming distance measures the mismatches, regardless of the order, we believe that this method could be especially successful in combinatorial problems where the order of the elements in the vector is not as relevant as the absolute position of the items, such as the graph-partitioning problem [11].

Acknowledgments

This work was partially supported by Basque Government(Elkartek program), by the Spanish Government (BERC program 2018-2021, the project TIN2016-78365-R, the project PID2019-106453GA-I00, BCAM Severo Ochoa excellence accreditation SVP-2014-068574 and SEV-2013-0323), and by AEI/FEDER, UE (through the project TIN2017-82626-R).

References

[1] K. Anstreicher, N. Brixius, J.-P. Goux, and J. Linderoth. Solving large quadratic assignment problems on computational grids. *Mathematical Programming*, 91(3):563–588, 2002.

[2] E. Arza, J. Ceberio, A. Pérez, and E. Irurozki. Approaching the Quadratic Assignment Problem with Kernels of Mallows Models Under the Hamming Distance. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '19, pages 141–142, New York, NY, USA, 2019. ACM.

[3] P. Awasthi, A. Blum, O. Sheffet, and A. Vijayaraghavan. Learning mixtures of ranking models. In *Advances in Neural Information Processing Systems*, pages 2609–2617, 2014.

[4] M. Ayodele, J. McCall, and O. Regnier-Coudert. RK-EDA: A novel random key based estimation of distribution algorithm. In *International Conference on Parallel Problem Solving from Nature*, pages 849–858. Springer, 2016.

[5] M. Ayodele, J. McCall, O. Regnier-Coudert, and L. Bowie. A random key based estimation of distribution algorithm for the permutation flowshop scheduling problem. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 2364–2371. IEEE, 2017.

[6] E. Bengoetxea, P. Larranaga, I. Bloch, A. Perchant, and C. Boeres. Inexact graph matching by means of estimation of distribution algorithms. *Pattern Recognition*, 35(12):2867–2880, 2002.

[7] U. Benlic and J.-K. Hao. Breakout local search for the quadratic assignment problem. *Applied Mathematics and Computation*, 219(9):4800–4815, Jan. 2013.

[8] U. Benlic and J.-K. Hao. Memetic search for the quadratic assignment problem. *Expert Systems with Applications*, 42(1):584–595, 2015.

[9] P. A. Bosman and D. Thierens. Crossing the road to efficient ideas for permutation problems. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pages 219–226. Morgan Kaufmann Publishers Inc., 2001.

[10] N. W. Brixius and K. M. Anstreicher. The Steinberg wiring problem. In *The Sharpest Cut: The Impact of Manfred Padberg and His Work*, pages 293–307. SIAM, 2004.

[11] A. Buluç, H. Meyerhenke, I. Safro, P. Sanders, and C. Schulz. Recent advances in graph partitioning. In *Algorithm Engineering*, pages 117–158. Springer, 2016.

[12] R. E. Burkard, S. E. Karisch, and F. Rendl. Qaplib–a quadratic assignment problem library. *Journal of Global optimization*, 10(4):391–403, 1997.

[13] B. Calvo and G. Santafé Rodrigo. scmamp: Statistical comparison of multiple algorithms in multiple problems. *The R Journal, Vol. 8/1, Aug. 2016*, 2016.

[14] B. Calvo, O. M. Shir, J. Ceberio, C. Doerr, H. Wang, T. Bäck, and J. A. Lozano. Bayesian performance analysis for black-box optimization benchmarking. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '19, pages 1789–1797, New York, NY, USA, 2019. ACM.

[15] J. Ceberio. *Solving permutation problems with estimation of distribution algorithms and extensions thereof.* PhD thesis, Universidad del País Vasco-Euskal Herriko Unibertsitatea, 2014.

[16] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano. A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems. *Progress in Artificial Intelligence*, 1(1):103–117, 2012.

[17] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano. A distance-based ranking model estimation of distribution algorithm for the flowshop scheduling problem. *IEEE Transactions on Evolutionary Computation*, 18(2):286–300, 2014.

[18] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano. A Distance-Based Ranking Model Estimation of Distribution Algorithm for the Flowshop Scheduling Problem. *IEEE Transactions on Evolutionary Computation*, 18(2):286–300, Apr. 2014.

[19] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano. Extending distance-based ranking models in estimation of distribution algorithms. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 2459–2466. IEEE, 2014.

[20] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano. A review of distances for the Mallows and generalized Mallows estimation of distribution algorithms. *Computational Optimization and Applications*, 62(2):545–564, Nov 2015.

[21] J. Ceberio, A. Mendiburu, and J. A. Lozano. The Plackett-Luce ranking model on permutation-based optimization problems. In *2013 IEEE Congress on Evolutionary Computation*, pages 494–501, Cancun, Mexico, June 2013. IEEE.

[22] J. Ceberio, A. Mendiburu, and J. A. Lozano. Kernels of Mallows Models for Solving Permutation-based Problems. In *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference - GECCO '15*, pages 505–512, Madrid, Spain, 2015. ACM Press.

[23] J. Ceberio, A. Mendiburu, and J. A. Lozano. Kernels of Mallows models for solving permutation-based problems. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 505–512. ACM, 2015.

[24] J. Ceberio, R. Santana, A. Mendiburu, and J. A. Lozano. Mixtures of generalized Mallows models for solving the quadratic assignment problem. In *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pages 2050–2057. IEEE, 2015.

[25] J. S. De Bonet, C. L. Isbell Jr, and P. A. Viola. Mimic: Finding optima by estimating probability densities. In *Advances in neural information processing systems*, pages 424–430, 1997.

[26] T. Dokeroglu and A. Cosar. A novel multistart hyper-heuristic algorithm on the grid for the quadratic assignment problem. *Engineering Applications of Artificial Intelligence*, 52:10–25, June 2016.

[27] Z. Drezner. A new genetic algorithm for the quadratic assignment problem. *INFORMS Journal on Computing*, 15(3):320–330, 2003.

[28] Z. Drezner, P. M. Hahn, and É. D. Taillard. Recent advances for the quadratic assignment problem with special emphasis on instances that are difficult for meta-heuristic methods. *Annals of Operations research*, 139(1):65–94, 2005.

[29] M. Fischetti, M. Monaci, and D. Salvagnin. Three ideas for the quadratic assignment problem. *Operations research*, 60(4):954–964, 2012.

[30] M. A. Fligner and J. S. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society. Series B (Methodological)*, 48(3):359–369, 1986.

[31] L. M. Gambardella, É. D. Taillard, and M. Dorigo. Ant colonies for the quadratic assignment problem. *Journal of the operational research society*, 50(2):167–176, 1999.

[32] A. Haghani and M.-C. Chen. Optimizing gate assignments at airport terminals. 32:437–454, 08 1998.

[33] P. Hahn and J. Krarup. A hospital facility layout problem finally solved. 12, 05 2000.

[34] D. R. Hunter. Mm algorithms for generalized bradley-terry models. *Ann. Statist.*, 32(1):384–406, 02 2004.

[35] E. Irurozki, B. Calvo, and J. Lozano. PerMallows: An R package for Mallows and generalized Mallows models. *Journal of Statistical Software, Articles*, 71(12):1–30, 2016.

[36] E. Irurozki, B. Calvo, and J. A. Lozano. Mallows and Generalized Mallows model for matchings. *Bernoulli*, 25(2):1160–1188, 05 2019.

[37] T. James, C. Rego, and F. Glover. A cooperative parallel tabu search algorithm for the quadratic assignment problem. *European Journal of Operational Research*, 195(3):810 – 826, 2009.

[38] T. James, C. Rego, and F. Glover. A cooperative parallel tabu search algorithm for the quadratic assignment problem. *European Journal of Operational Research*, 195(3):810–826, June 2009.

[39] T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 184–192, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[40] J. D. Knowles and D. Corne. Towards landscape analyses to inform the design of hybrid local search for the multiobjective quadratic assignment problem. *HIS*, 87:271–279, 2002.

[41] T. Koopmans and M. J. Beckmann. Assignment problems and the location of economic activities. Cowles Foundation Discussion Papers 4, Cowles Foundation for Research in Economics, Yale University, 1955.

[42] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña. Optimization in continuous domains by learning and simulation of gaussian networks. 2000.

[43] P. Larrañaga and J. A. Lozano. *Estimation of distribution algorithms: A new tool for evolutionary computation*, volume 2. Springer Science & Business Media, 2001.

[44] E. L. Lawler. The quadratic assignment problem. *Management science*, 9(4):586–599, 1963.

[45] G. Lebanon and Y. Mao. Non-parametric modeling of partially ranked data. *Journal of Machine Learning Research*, 9(Oct):2401–2429, 2008.

[46] H. Liu, A. Abraham, and J. Zhang. A particle swarm approach to quadratic assignment problems. In *Soft computing in industrial applications*, pages 213–222. Springer, 2007.

[47] E. M. Loiola, N. M. M. de Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido. A survey for the quadratic assignment problem. *European Journal of Operational Research*, 176(2):657–690, Jan. 2007.

[48] C. L. Mallows. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957.

[49] P. Merz and B. Freisleben. Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE transactions on evolutionary computation*, 4(4):337–352, 2000.

[50] P. Mickey R. Wilhelm Ph.D. and P. Thomas L. Ward Ph.D. Solving quadratic assignment problems by 'simulated annealing'. *IIE Transactions*, 19(1):107–119, 1987.

[51] H. D. Mittelmann and D. Salvagnin. On solving a hard quadratic 3-dimensional assignment problem. *Mathematical Programming Computation*, 7(2):219–234, June 2015.

[52] T. B. Murphy and D. Martin. Mixtures of distance-based models for ranking data. *Computational statistics & data analysis*, 41(3-4):645–655, 2003.

[53] P. M. Pardalos, H. Wolkowicz, et al. *Quadratic Assignment and Related Problems: DIMACS Workshop, May 20-21, 1993*, volume 16. American Mathematical Soc., 1994.

[54] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[55] R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202, 1975.

[56] T. Pradeepmon, V. V. Panicker, and R. Sridharan. Hybrid estimation of distribution algorithms for solving a keyboard layout problem. *Journal of Industrial and Production Engineering*, 35(6):352–367, Aug. 2018.

[57] C. S. Rabak and J. S. Sichman. Using a-teams to optimize automatic insertion of electronic components. *Advanced Engineering Informatics*, 17(2):95–106, 2003.

[58] S. Sahni and T. Gonzalez. P-complete approximation problems. *J. ACM*, 23(3):555–565, July 1976.

[59] J. Skorin-Kapov. Tabu search applied to the quadratic assignment problem. *ORSA Journal on computing*, 2(1):33–45, 1990.

[60] N. J. A. Sloane. The On-Line Encyclopedia of Integer Sequences. A000166. Subfactorial or rencontres numbers, or derangements: number of permutations of n elements with no fixed points.

[61] É. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel computing*, 17(4-5):443–455, 1991.

[62] D. M. Tate and A. E. Smith. A genetic approach to the quadratic assignment problem. *Computers and Operations Research*, 22(1):73–84, 1995.

[63] S. Tsutsui, M. Pelikan, and D. E. Goldberg. Using edge histogram models to solve permutation problems with probabilistic model-building genetic algorithms. *IlliGAL Report*, 2003022, 2003.

[64] S. Tsutsui, M. Pelikan, and D. E. Goldberg. Node histogram vs. edge histogram: a comparison of pmbgas in permutation domains. *MEDAL Report*, (2006009), 2006.

[65] M. Zangari, A. A. Constantino, and J. Ceberio. A decomposition-based kernel of Mallows models algorithm for bi-and tri-objective permutation flowshop scheduling problem. *Applied Soft Computing*, 71:526–537, 2018.

[66] Q. Zhang, J. Sun, E. Tsang, and J. Ford. Combination of guided local search and estimation of distribution algorithm for quadratic assignment problems. In *Proceedings of the Genetic and Evolutionary Computation Conference, Chicago, IL, USA*, pages 42–48, 2003.