A Random Key based Estimation of Distribution Algorithm for the Permutation Flowshop Scheduling Problem

Mayowa Ayodele, John McCall, Olivier Regnier-Coudert, Liam Bowie
Robert Gordon University
Aberdeen, Scotland
Email: {m.m.ayodele,j.mccall,o.regnier-coudert, l.r.bowie}@rgu.ac.uk

Abstract—**Random Key (RK) is an alternative representation for permutation problems that enables application of techniques generally used for continuous optimisation. Although the benefit of RKs to permutation optimisation has been shown, its use within Estimation of Distribution Algorithms (EDAs) has been a challenge. Recent research proposing a RK-based EDA (RK-EDA) has shown that RKs can produce competitive results with state of the art algorithms. Following promising results on the Permutation Flowshop Scheduling Problem, this paper presents an analysis of RK-EDA for optimising the total flow time. Experiments show that RK-EDA outperforms other permutation-based EDAs on instances of large dimensions. The difference in performance between RK-EDA and the state of the art algorithms also decreases when the problem difficulty increases.**

I. Introduction

Many real-world scheduling problems can be naturally represented as permutations and so there has been a lot of research on solving permutation problems in the evolutionary computation community. Some of the most widely studied permutation problems are Quadratic Assignment Problem (QAP), Linear Ordering Problem (LOP), Travelling Salesman Problems (TSP) and Permutation Flowshop Scheduling Problem (PFSP). In recent years, there has particularly been a lot of research interest in solving the PFSP, using various objective functions [1], [2], [3], [4], [5].

Modelling the space of permutations is generally considered a difficult task and has been a challenging area for Estimation of Distribution Algorithms (EDAs) whose results did not match those of state of the art methods [6]. This was attributed to the fact that a large proportions of EDAs were based on concepts borrowed from the integer and continuous optimisation domains rather than using characteristics of the permutation spaces. The need to respect variable mutual exclusivity is a key challenge to the modelling of permutations [7], [8], which often results in the use of additional procedures or alternative representations. Random Keys (RKs) have been used in several continuous-based EDAs [6]. The RK representation uses continuous values that can be ranked to order items into a permutation, without expensive translation mechanism. However, EDAs using RKs generally exhibit poor results [6].

A limitation of the RK representation is that several distinct RKs may produce similar permutations, introducing redundancies and plateaux in the search space [9], [10]. Variability in the values that capture the same priority across solutions of a population has also been mentioned as an issue of RKs that limits the information captured by probabilistic models and affect the quality of their results [11]. Models that are more specific to permutations such as histogram models [12], [13], permutation distribution models [3], [14], [7] and factoradics [15] showed better performances. RKs however have an advantage over most other permutation representations as they always produce permutation feasible solutions [16]. The problem with most RK based EDAs is that the representation has not been sufficiently adapted for the permutation domain.

Some examples of RK based EDAs are REDA [17], EGNA$_{ee}$ & UMDA$_c$ [18]. REDA uses the triangulation of Bayesian network approach and focuses on model efficiency by modelling subset nodes of a problem. EGNA$_{ee}$ builds a Gaussian network where the structure of a problem is learnt using edge exclusion tests [18]. The UMDA$_c$ which is also a structure identification algorithm based on Gaussian network performs hypothesis tests to identify the density of its model's components. In addition, IDEA-ICE [9] can also be classified as a RK based EDA, although it uses a crossover operator to preserve building blocks in addition to its probabilistic model. To increase the chances of properly combining the building blocks, RKs associated with the building blocks are rescaled. The IDEA-ICE showed better performance compared to the RK based EDAs.

In [19], a novel Random Key based Estimation of Distribution Algorithm (RK-EDA) was proposed. RK-EDA is a light weight univariate EDA whose probabilistic model, similar to UMDA$_c$, is based on mean values of genes in more promising solutions of a population. It exploits already found good genes by sampling a Gaussian distribution based on mean and variance values. Unlike UMDA$_c$, RK-EDA imposes a user defined variance parameter rather than a population generated one. Furthermore, RK-EDA uses a cooling scheme to automatically control the defined variance value which creates an exploration-exploitation balance in the algorithm. To limit the effect of redundancies introduced by the RK representation, RK-EDA normalises its RKs. These changes made a significant difference making RK-EDA more competitive than other EDAs that use the RK representation on TSP, LOP and in particular on PFSP using makespan. The research in [19] also suggested that RK-EDA is competitive with other leading EDAs such as Node Histogram Based Sampling Algorithm (NHBSA) [12], Edge Histogram Based Sampling Algorithm (EHBSA) [13] and Generalised Mallows model based EDA (GM-EDA) [3].

RK-EDA was shown to be particularly promising on PFSP in [19] suggesting that RK-EDA may scale well to larger instances of the problem. In this paper, we apply RK-EDA to a wider range of the well-known Taillard's PFSP instances [20]. Results presented are based on Total Flow Time (TFT) optimisation criterion. This paper not only compares RK-EDA with leading EDAs but also other state-of-the-art algorithms. We are able to show the scalability of RK-EDA on the largest PFSP instances.

The rest of this paper is structured as follows. Section II formally defines the PFSP and also presents a review of some of the most competitive algorithms at solving the PFSP. Section III describes RK-EDA. Section IV presents the experimental settings while Section V presents some results and evaluation. Conclusions are presented in Section VI.

II. Permutation Flowshop Scheduling Problem

The PFSP consists of a set of jobs indexed 1, ..., n to be scheduled on a set of m machines. Each job has m operations to be performed by all m machines where the j^{th} operation of each job must be performed by machine j. A job can have its j^{th} operation performed once its $j-1^{th}$ operation has been completed by machine $j-1$ and machine j is available. The objective of this problem is to find a sequence of jobs that minimises the sum of times that each job remains on the flowshop known as the TFT. This objective function was chosen in order to compare results with those of most recent algorithms. Another common objective is to minimise makespan and we refer the reader to [19] for more details and additional results of RK-EDA. The TFT of an ordering of jobs π is formally defined as follows.

$$F(\pi) = \sum_{i=0}^{n} C_{\pi(i),m} \tag{1}$$

In eq. (1), $C_{\pi(i),m}$ denotes the completion time of a job ranked $\pi(i)$ on machine m and is calculated as follows.

$$C_{\pi(i),j} = \begin{cases} P_{\pi(i),j} & i = j = 1 \\ P_{\pi(i),j} + C_{\pi(i-1),j} & i > 1,\ j = 1 \\ P_{\pi(i),j} + C_{\pi(i),j-1} & i = 1,\ j > 1 \\ P_{\pi(i),j} + Max(C_{\pi(i-1),j}, C_{\pi(i),j-1}) & i > 1,\ j > 1 \end{cases} \tag{2}$$

In eq. (2), we denote the processing time required to perform a job ranked $\pi(i)$ on machine j by $P_{\pi(i),j}$.

The review in [6] presents a range of EDAs applied to the PFSP as well as other common permutation problems. The most competitive results were obtained by the histogram based sampling algorithms, NHBSA [12] and EHBSA [13]. NHBSA models node frequencies at each position while EHBSA models the relative relation between two nodes in solutions of a population [13]. These algorithms directly encodes permutations thereby needing no additional translation procedure. They also do not require repair procedures.

Another competitive EDA proposed for solving the PFSP is based on the Generalised Mallows model called GM-EDA [3]. GM-EDA is the first main attempt at solving permutation problems based on parametric models. It uses a distance-based exponential probabilistic model over the space of permutations. GM-EDA was able to outperform the histogram based sampling algorithms on many large problem instances. A hybrid of GM-EDA with Variable Neighbourhood Search (VNS) called HGM-EDA , which enhances the performance of GM-EDA, was also proposed in [3]. Both GM-EDA and HGM-EDA use a restart mechanism where the population is regenerated once all solutions in the population become the same [3].

Apart from GM-EDA and HGM-EDA, the Asynchronous Genetic local search Algorithm (AGA) in [1] and Differential Evolution Algorithm for Permutation (DEP) [5] also make use of restart mechanisms and local search (LS) methods to improve their search. These are some of the most competitive meta-heuristics applied to the PFSP. In addition to these, some LS approaches have presented even more competitive results on certain instances of the Taillard's problem set. They are the VNS$_4$ [21], Iterated Greedy Algorithm (IGA) [2] and Iterated Local Search (ILS) [2].

III. RK-EDA

In this section, we present the algorithmic details of RK-EDA.

Algorithm 1 RK-EDA

```
1: Initialise $\sigma$, $t_s$ and $p_s$
2: Generate initial population $P$ of size $p_s$
3: for $g = 1$ to $MaxGen$ do
4:     Evaluate and rescale individuals in $P$
5:     Select best $t_s < p_s$ solutions to form $S$
6:     Calculate $\mu = \{\mu_1, \mu_2, \cdots, \mu_n\}$
7:     $c = 1 - \frac{g}{MaxGen}$
8:     $\sigma_g = \sigma * c$
9:     $M = N(\mu, \sigma_g)$
10:    $P_{new} = \emptyset$
11:    repeat
12:        Sample $M$ to generate offspring off
13:        Add off to $P_{new}$
14:    until $|P_{new}| = p_s$
15:    $P = P_{new}$
16: end for
```

As shown in Alg. 1, RK-EDA requires the initialisation of three parameters which are initial variance σ, truncation size t_s and population size p_s. Since the stopping criteria is based on the number of fitness evaluations allowed (*FEs*), the maximum number of generations $MaxGen$ is estimated by dividing *FEs* by p_s.

A population P of RKs is randomly generated, evaluated and rescaled. For the rescaling procedure to be carried out, the RKs are converted to ranks as illustrated in Figure 1 for the RKs [0.12, 0.57, 0.23, 0.25, 0.99]. The ranks are then rescaled to values between int the [0,1] interval. This is done by setting $rescaledRK_i = \frac{rank_i - 1}{n-1}$ where $rescaledRK_i$ and $rank_i$ are respectively the rescaled RK and rank of gene i, and n is the problem size. Based on this approach, a distinct set of RKs (for example [0.01, 0.06, 0.03, 0.04, 0.2]) representing the same permutation will have the same rescaled RK value. This approach minimises redundancy and improves the information captured by the probabilistic model.

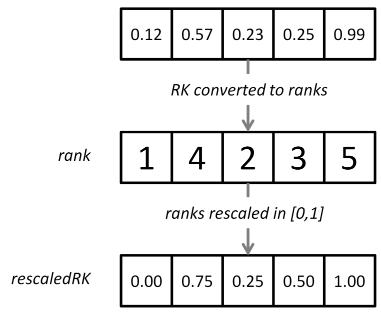

Fig. 1. RK rescaling

Once rescaled and evaluated, the best t_s solutions in P are selected to generate a population of promising solutions S. The mean of all RKs at each index $\{1, \cdots, n\}$ is computed from S. μ is the vector of all indices' mean μ_i.

Furthermore, the cooling rate c is calculated with respect to the algorithm's current generation such that its value is higher at the start of the search and low at the end. The rate c is used to generate generational variance σ_g. Multiplying c with σ to form σ_g makes it possible to achieve higher exploration at the start of the run and more exploitation as g increases.

The probabilistic model M saves μ, which comprises of n values as well as σ_g, which is a single value. M is therefore of size $n+1$ and is defined as a normal distribution $N(\mu, \sigma_g)$. M is updated at the end of each generation g.

An offspring solution *off* is generated by sampling M. Each gene *off*$_i$ is generated by sampling $N(\mu_i, \sigma_g)$. *off* is repeatedly added to the offspring population P_{new} until its size equals p_s. At the end of each generation, P_{new} completely replaces the parent population P. Note that although *off*$_i$ obtained by sampling M may be outside the $[0,1]$ interval, it does not represent an issue as it does not prevent the RKs to be ranked and rescaled.

IV. Experimental Settings

In this section, we present the test sets and parameter settings for RK-EDA.

A. Problem sets

To be able to assess the performance of RK-EDA on difference sizes of the PFSP, we apply RK-EDA to the Taillard's benchmark problems [20]. The following problem sets were considered.

1) Size 20: 20×5, 20×10, 20×20,
2) Size 50: 50×5, 50×10, 50×20,
3) Size 100: 100×5, 100×10, 100×20,
4) Size 200: 200×10, 200×20 and
5) Size 500: 500×20

For the main experimental analysis, RK-EDA is compared with other algorithms using the first two instances of each problem set (sizes 20 - 200). In order to gain insight on the performance of RK-EDA on large dimensions, additional runs were carried out on the 10 largest instances, that is the 500×20 problems to understand how RK-EDA scales to difficult problems.

RK-EDA is run 20 times for each instance and performance measured using Average Relative Percentage Deviation (ARPD) from best and is calculated as shown in Eq. 3.

$$ARPD = (\sum_{i=0}^{20} \frac{(Algorithm_i - Best) \times 100}{Best}) \div 20 \qquad (3)$$

As discussed in Section II, RK-EDA is compared to leading EDAs, which are GM-EDA, NHBSA and EHBSA based on the results gathered in [3]. Performance is also compared with other leading algorithms such as ILS, IGA, AGA, VNS$_4$ and HGM-EDA based on the results presented in [5]. For all comparisons, statistical significance (95% confidence interval) is measured by means of unpaired t-test.

B. Parameter Setting

TABLE I
Parameter Values for RK-EDA

Parameters	Values
Population Size (p_s)	$10n$
Truncation Size (t_s)	$0.1{*}p_s$
Variance (σ)	0.15
Maximum Number of Generations ($MaxGen$)	FEs/p_s
Number of Runs	20

TABLE II
Stopping Criteria: Number of Fitness Evaluations

Problem Sizes	Fitness Evaluations
20×05	182,224,100
20×10	224,784,800
20×20	256,896,400
50×05	220,712,150
50×10	256,208,100
50×20	275,954,150
100×5	235,879,800
100×10	266,211,000
100×20	283,040,000
200×10	272,515,500
200×20	287,728,850

To be able to compare with the leading EDAs, we use similar parameters as [3] except that we do not consider elitism. This is because preliminary experiments show that elitism does not improve the performance of RK-EDA. The parameter values are presented in Table I. These values include a population size of $10n$ and 10% of it as truncation size. We set the initial variance to 0.15 across all problems.

Table II shows the number of fitness evaluations used by RK-EDA for problem instances of dimensions 20 to 200. This budget matches the one used by the algorithms included in the comparison. Note that only 20% of this budget was used for instances of dimension 500 (52,063,350 evaluations), used to gain insight on the scalability of RK-EDA.

V. Results and Discussion

In this section, we present the results obtained by RK-EDA on the selected PFSP problem instances. Results are first compared with those of other permutation-based EDAs before being compared to those of the leading algorithms on PFSP. Finally, we investigate whether mechanisms inherent to RK-EDA such as its cooling scheme are able to maintain diversity throughout the search without requiring the use of LS.

A. Results

The ARPDs of RK-EDA on the 22 selected problem instances are presented in Table III, along with the ARPD of all other algorithms considered in the comparison. The TFTs for the best known solutions used to compute the ARPD values are also given for reference.

Table IV shows results obtained on RK-EDA on the 10 instances of size 500 using a fifth of the budget allocated to other algorithms. In Tables III and IV, ARPDs that are significantly worse than that of RK-EDA are presented in italics while those that are significantly better are presented in bold.

B. Comparing RK-EDA with stand-alone EDAs

The first objective of this study is to assess the performance of RK-EDA with respect to other EDAs. For this purpose, we focus on the three standalone EDAs that are NHBSA, EHBSA and GM-EDA and do not consider hybrid methods, such as HGM-EDA.

As shown in Table III, the histogram model based EDAs (NHBSA and EHBSA) are able to reach the best known total flow time on the smaller problems. They also perform better than the GM-EDA or RK-EDA on these problems. However, as the dimension of the problems increases, they become less competitive. GM-EDA and RK-EDA present better results on the larger problems.

Although RK-EDA is significantly worse than the NHBSA on the first eleven problems, it is significantly better on the other half which are the larger ones. A similar trait is exhibited on the EHBSA where RK-EDA is significantly better on the largest 8 of the 22 instances considered in Table III. However, there is no clear winning algorithm between GM-EDA and RK-EDA on the smaller problems. This is because RK-EDA is sometimes significantly better than GM-EDA and vice versa while other times their results are not statistically different. RK-EDA is however steadily significantly better than the GM-EDA on problems of size 200. In general, RK-EDA presents the best results of the four EDAs on some of size 100 problems and all of size 200 problems.

Futhermore, on the large instances (of size 500), RK-EDA significantly outperforms all of the EDAs, despite using much less evaluations. On those instances, the ARPD of RK-EDA is between 7 and 26 times lower than NHBSA, between 9 and 30 times lower than EHBSA and between 9 and 32 times lower than GM-EDA. Although these results are only preliminary, RK-EDA appears as the most suitable EDA for large PFSP problems. In [3], the authors suggest that GM-EDA may suffer from slow convergence on large dimensions, an issue avoided by RK-EDA and its cooling scheme as discussed in Section V-D.

C. Comparing RK-EDA with Leading Algorithms

The other algorithms included in the study are HGM-EDA, AGA, VNS$_4$, ILS, IGA and DEP. Among those, DEP exhibits the best ARPD on problems of dimensions 200 or lower. On problems of similar sizes, HGM-EDA shows good results and shows that when combined with LS procedures, EDAs can be competitive with other types of algorithms on PFSP. RK-EDA is able to significantly outperform the VNS$_4$ on one of the size 100 instances and all of the size 200 instances considered. Although RK-EDA is significantly worse than HGM-EDA, AGA, ILS, IGA and DEP on problems of size 200 and lower, its performance is more comparable as problem size becomes bigger.

Figure 2 shows the ratio between the ARPD of RK-EDA and the best algorithm on each instance. It also shows the ratio between the ARPD of the best EDA on each instance and the best algorithm on each instance. Despite variations and being significantly outperformed by the best algorithms, RK-EDA's performance tends to get closer to those of the best methods as the dimension increases. A similar trend is observed on the performance gap between the best EDAs and the best algorithms.

In order to strengthen the idea that RK-EDA performs particularly well on large problems, it was run for a small number of evaluations on the dimension 500 instances. The results gathered in Table IV shows that on the largest PFSP instances, RK-EDA significantly outperforms HGM-EDA on all instances despite not using any LS procedure. This demonstrates the ability of the proposed RK model to efficiently guide the search. It also significantly outperforms the DEP and VNS$_4$ on all instances of size 500. Although it does significantly worse than the AGA on the second instance of the size 500 problem, it is significantly better on four others and not significantly different on the rest.

Figure 3 highlights how RK-EDA performs with respect to the best algorithms on the large instances (ILS, IGA and DEP). It's better performance compared to the DEP corresponds to the relative difficulties exhibited by DEP on larger problems

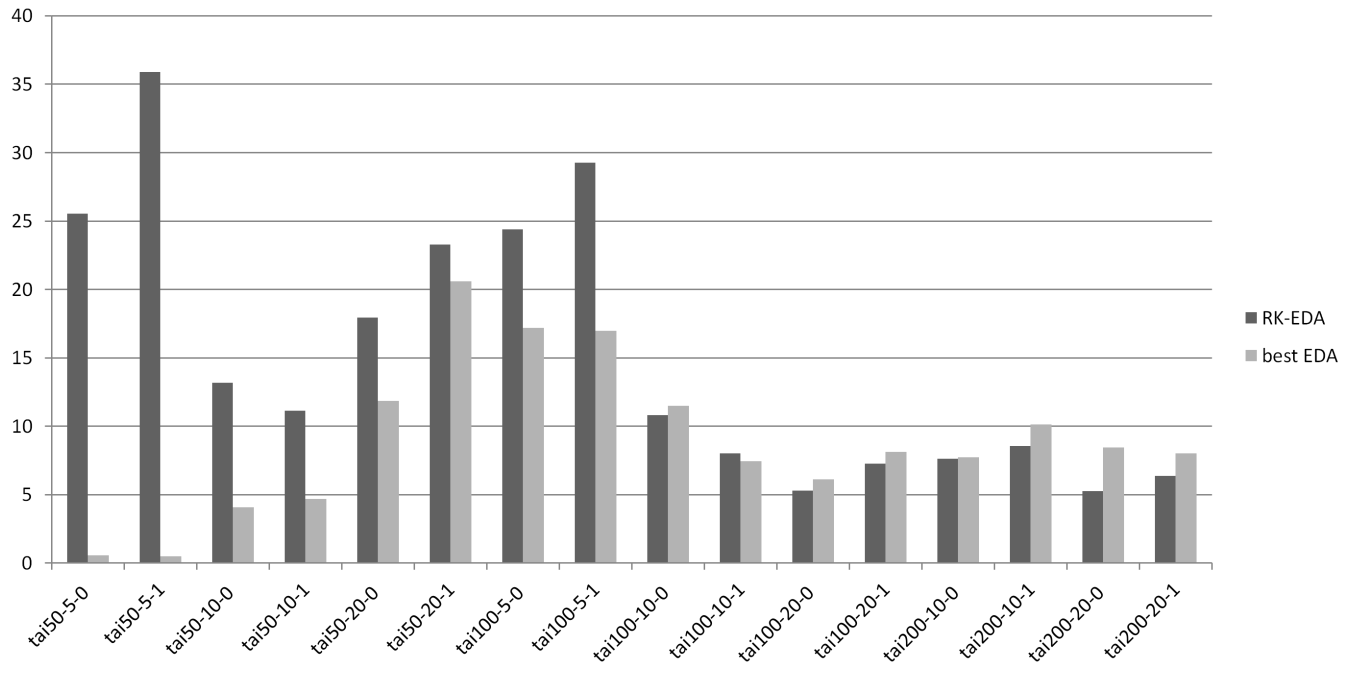

Fig. 2. Ratio between ARPD of RK-EDA and best algorithm; and best EDA and best algorithm

TABLE III
ARPD RESULTS FOR FIRST TWO INSTANCES OF TAILLARD'S BENCHMARK SETS 20×5 - 200×20

Problems	Best Known TFT	RK-EDA	GM-EDA	EHBSA	NHBSA	HGM-EDA	AGA	VNS_4	ILS	IGA	DEP
tai20-5-0	14033	0.37	**0.18**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
tai20-5-1	15151	0.48	0.48	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
tai20-10-0	20911	0.44	0.45	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
tai20-10-1	22440	0.98	**0.54**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
tai20-20-0	33623	0.22	*0.65*	*0.46*	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
tai20-20-1	31587	0.28	0.28	**0.01**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
tai50-5-0	64803	1.30	**0.79**	**0.03**	**0.79**	**0.12**	**0.05**	**0.78**	**0.05**	**0.07**	**0.05**
tai50-5-1	68062	2.06	**0.94**	**0.03**	**0.91**	**0.12**	**0.06**	**0.88**	**0.13**	**0.14**	**0.08**
tai50-10-0	87204	2.33	**2.11**	**0.72**	**1.86**	**0.39**	**0.33**	**1.12**	**0.44**	**0.40**	**0.18**
tai50-10-1	82820	2.46	2.45	**1.03**	**1.97**	**0.60**	**0.22**	**1.09**	**0.59**	**0.39**	**0.30**
tai50-20-0	125831	1.81	1.76	**1.20**	**1.75**	**0.39**	**0.10**	**0.65**	**0.33**	**0.41**	**0.14**
tai50-20-1	119247	1.11	*1.59*	**0.98**	*1.75*	**0.23**	**0.05**	**0.52**	**0.14**	**0.15**	**0.07**
tai100-5-0	253605	1.23	**0.87**	**1.08**	*1.75*	**0.23**	**0.29**	1.25	**0.48**	**0.46**	**0.05**
tai100-5-1	242579	1.51	**1.08**	**0.88**	*1.83*	**0.35**	**0.30**	*1.80*	**0.80**	**0.82**	**0.05**
tai100-10-0	299101	1.70	*1.80*	*2.80*	*2.60*	**0.44**	**0.43**	**1.63**	**0.42**	**0.56**	**0.16**
tai100-10-1	274566	2.25	**2.08**	*3.17*	*3.03*	**0.69**	**0.60**	**1.58**	**0.54**	**0.85**	**0.28**
tai100-20-0	366438	1.96	*2.26*	*4.55*	*3.12*	**0.67**	**0.80**	**1.70**	**0.79**	**0.71**	**0.37**
tai100-20-1	373138	1.82	*2.04*	*4.13*	*3.17*	**0.58**	**0.55**	**1.43**	**0.54**	**0.73**	**0.25**
tai200-10-0	1047541	1.18	1.20	*5.03*	*4.06*	**0.19**	**0.49**	*1.26*	**0.29**	**0.15**	**0.22**
tai200-10-1	1035783	1.25	*1.49*	*5.89*	*4.65*	**0.32**	**0.94**	*1.54*	**0.71**	**0.65**	**0.15**
tai200-20-0	1225282	1.07	*1.72*	*6.26*	*5.24*	**0.39**	**0.76**	*1.48*	**0.30**	**0.40**	**0.20**
tai200-20-1	1239246	1.32	*1.66*	*6.98*	*5.59*	**0.54**	**1.07**	*1.67*	**0.64**	**0.74**	**0.21**

TABLE IV
ARPD RESULTS FOR TAILLARD'S BENCHMARK SET 500 × 20

Instances	Best Known TFT	RK-EDA	GM-EDA	EHBSA	NHBSA	HGM-EDA	AGA	VNS_4	ILS	IGA	DEP
tai500_20_0	6687476	0.29	*9.24*	*8.87*	*7.60*	*2.34*	*0.42*	*0.66*	**0.13**	**0.22**	*1.32*
tai500_20_1	6809182	0.68	*8.91*	*8.54*	*7.20*	*2.25*	**0.55**	0.69	**0.24**	**0.36**	*0.96*
tai500_20_2	6734895	0.47	*8.66*	*8.88*	*7.46*	*2.23*	0.42	*0.59*	**0.14**	**0.13**	*1.25*
tai500_20_3	6767341	0.63	*9.07*	*8.55*	*7.31*	*2.18*	0.56	*0.74*	**0.16**	**0.19**	*1.13*
tai500_20_4	6720679	0.98	*9.28*	*8.92*	*7.54*	*2.45*	0.91	0.93	**0.15**	**0.17**	*1.26*
tai500_20_5	6723390	0.60	*9.04*	*9.44*	*7.78*	*2.56*	0.60	*0.84*	**0.14**	**0.18**	*0.73*
tai500_20_6	6681650	0.52	*9.59*	*9.20*	*7.68*	*2.46*	*0.68*	*0.86*	**0.17**	**0.21**	*1.34*
tai500_20_7	6769821	0.42	*8.62*	*8.88*	*7.40*	*2.09*	*0.31*	*0.58*	**0.28**	**0.20**	0.73
tai500_20_8	6701696	0.38	*8.99*	*8.64*	*7.37*	*2.19*	*0.43*	*0.74*	**0.10**	**0.12**	*1.24*
tai500_20_9	6755620	0.32	*8.70*	*8.63*	*7.35*	*2.18*	0.37	*0.62*	**0.15**	**0.10**	*1.04*

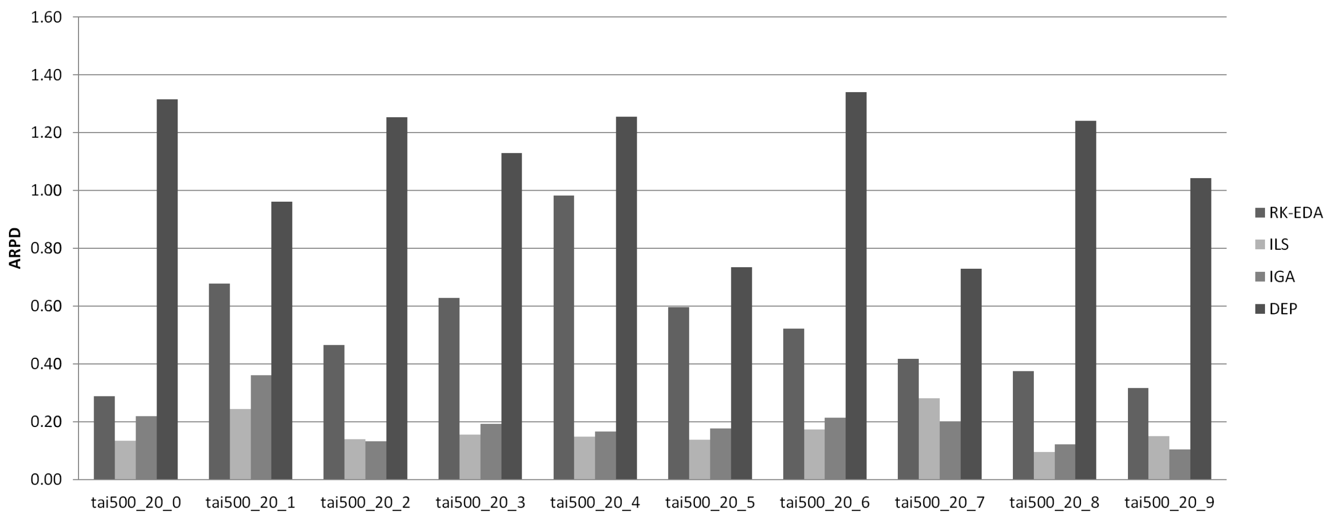

Fig. 3. ARPD of RK-EDA and best algorithms on dimension 500 instances

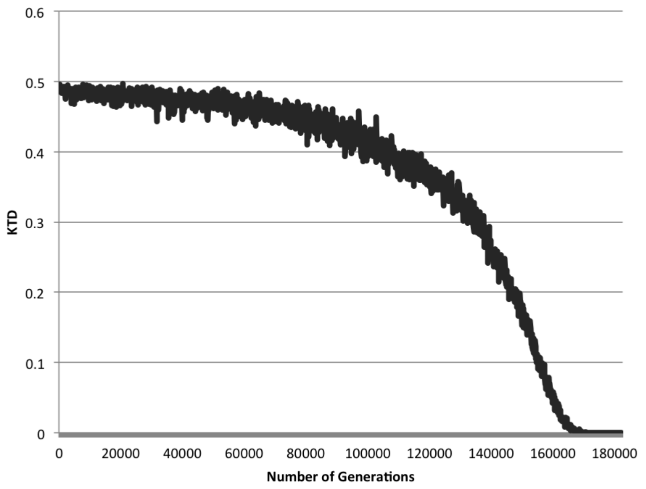

Fig. 4. Measure of KTD on tai20_5_0

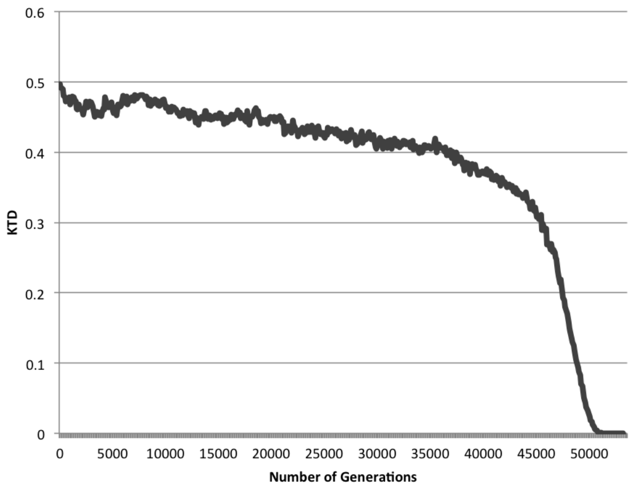

Fig. 5. Measure of KTD on tai100_10_0

due to slow convergence [5]. However, RK-EDA fails to match the level of performance of ILS and IGA, both very robust on those instances. ILS and IGA are the algorithms that scale the best and thus present the best results on problems of size 500. It is important to note however that in this setting, RK-EDA used 80% fewer fitness evaluations than its counterparts.

D. Diversity in RK-EDA

Since RK-EDA does not use restart mechanisms or LS methods, it is important to ensure that the algorithm allows the search to both explore and exploit the space. To understand the behaviour of RK-EDA, the diversity in this population is measured. For this purpose, the Kendal Tau Distance (KTD) [22] is often used. It represents the number of adjacent transposition that needs to be performed to transform a permutation into another one.

Pairwise KTD between two permutations π_1 and π_2 is calculated following eq. 4. The elements i and j belong to the set P of unordered pairs of elements obtained from π_1 and π_2. $K_{ij}(\pi_1, \pi_2)$ equals either 0 if i and j are in the same order, or 1 if i and j are in a different order in π_1 and π_2

$$KTD(\pi_1, \pi_2) = \sum_{\{i,j\} \in P} K_{ij}(\pi_1, \pi_2) \tag{4}$$

We calculate the KTD for all pairwise comparison from all solution pairs in a population at each generation. These are summed up and averaged by the number of pairwise comparison. To normalise the average KTD, we divide it by $n(n-1)/2$ where n is the problem size. This way a value of 1 indicates maximum disagreement while 0 indicates two identical permutations.

Figures 4 and 5 shows how KTD varies as the search progressed based on two runs of RK-EDA on tai20_5_0 and tai100_10_0 respectively. Both figures show a similar pattern despite being based on different dimensions. RK-EDA generally maintains a KTD above 0.4 for the first half of the search and exhibit a strong decline in KTD in the last part of the search to reach a state where all solutions are similar in the population. The pattern displayed corresponds to the intended behaviour of RK-EDA and of many algorithms in general by allowing a long exploration phase before exploiting the most promising areas of the search space.

The cooling scheme helps to maintain diversity in the population until its closer to the end of the search. This is a major difference between RK-EDA and many existing approaches [3], [5] that need restart mechanism to handle premature convergence.

VI. Conclusions

In this paper, we have presented an EDA that uses a light weight univariate model based on RKs. The light weight characteristics of the model is of particular importance when considering that learning a probability structure is considered the most expensive operation in EDAs [9]. One of the key feature of RK-EDA is its cooling scheme which manages the variance used in the model. Diversity computed experimentally confirmed that RK-EDA avoid premature convergence and exhibits both exploration and exploitation behaviours during the search.

RK-EDA produces very competitive results. It outperforms leading EDAs and several other methods on the largest PFSP instances. RK-EDA also presents better results than GM-EDA, an hybrid EDA, without using any LS procedure. Although RK-EDA is not as efficient as the best algorithms for PFSP, especially ILS and IGA, the difference between them tends to decrease significantly as the problem dimension grows.

The present paper highlights the importance of the cooling scheme and the variance parameter to manage diversity during the search. It is thus important that a systematic comparison of different cooling procedures and variance settings be carried out. Future work should also investigate strategies to improve RK-EDA's results on small instances and propose hybrid algorithms to enhance the results using LS.

References

[1] X. Xu, Z. Xu, and X. Gu, "An asynchronous genetic local search algorithm for the permutation flowshop scheduling problem with total flowtime minimization," *Expert systems with Applications*, vol. 38, no. 7, pp. 7970–7979, 2011.

[2] Q.-K. Pan and R. Ruiz, "Local search methods for the flowshop scheduling problem with flowtime minimization," *European Journal of Operational Research*, vol. 222, no. 1, pp. 31–43, 2012.

[3] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano, "A distance-based ranking model estimation of distribution algorithm for the flowshop scheduling problem," *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 2, pp. 286–300, 2014.

[4] V. Santucci, M. Baioletti, and A. Milani, "A differential evolution algorithm for the permutation flowshop scheduling problem with total flow time criterion," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2014, pp. 161–170.

[5] ——, "Algebraic differential evolution algorithm for the permutation flowshop scheduling problem with total flowtime criterion," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 682–694, 2016.

[6] 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*, vol. 1, no. 1, pp. 103–117, 2012.

[7] J. Ceberio, A. Mendiburu, and J. A. Lozano, "The plackett-luce ranking model on permutation-based optimization problems," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 494–501.

[8] 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*. IEEE, 2014, pp. 2459–2466.

[9] P. A. Bosman and D. Thierens, "Crossing the road to efficient ideas for permutation problems," in *Proceedings of the 6th annual conference on Genetic and evolutionary computation*. ACM, 2001, pp. 219–226.

[10] M. Pelikan, S. Tsutsui, and R. Kalapala, "Dependency trees, permutations, and quadratic assignment problem," in *Genetic And Evolutionary Computation Conference: Proceedings of the 9 th annual conference on Genetic and evolutionary computation*, vol. 7, no. 11, 2007, pp. 629–629.

[11] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111–128, 2011.

[12] S. Tsutsui, "Probabilistic model-building genetic algorithms in permutation representation domain using edge histogram," in *Parallel Problem Solving from Nature, PPSN VII*. Springer, 2002, pp. 224–233.

[13] S. Tsutsui, M. Pelikan, and D. E. Goldberg, "Node histogram vs. edge histogram: a comparison of pmbgas in permutation domains," *MEDAL Report*, no. 2006009, 2006.

[14] 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*. ACM, 2015, pp. 505–512.

[15] O. Regnier-Coudert and J. McCall, "Factoradic representation for permutation optimisation," in *Parallel Problem Solving from Nature, PPSN XIII*. Springer, 2014, pp. 332–341.

[16] J. C. Bean, "Genetic algorithms and random keys for sequencing and optimization," *ORSA journal on computing*, vol. 6, no. 2, pp. 154–160, 1994.

[17] T. Romero and P. Larrañaga, “Triangulation of bayesian networks with recursive estimation of distribution algorithms,” *International Journal of Approximate Reasoning*, vol. 50, no. 3, pp. 472–484, 2009.

[18] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer, 2002, vol. 2.

[19] 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*. Springer, 2016, pp. 849–858.

[20] E. Taillard, “Benchmarks for basic scheduling problems,” *european journal of operational research*, vol. 64, no. 2, pp. 278–285, 1993.

[21] W. E. Costa, M. C. Goldbarg, and E. G. Goldbarg, “New vns heuristic for total flowtime flowshop scheduling problem,” *Expert Systems with Applications*, vol. 39, no. 9, pp. 8149–8161, 2012.

[22] M. A. Fligner and J. S. Verducci, “Multistage ranking models,” *Journal of the American Statistical association*, vol. 83, no. 403, pp. 892–901, 1988.