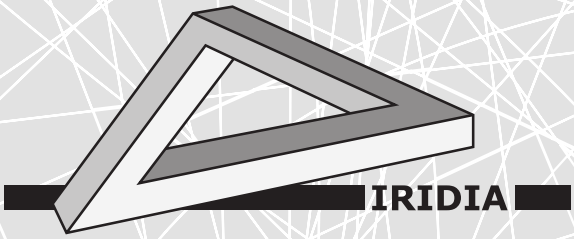


**Université Libre de Bruxelles**

*Institut de Recherches Interdisciplinaires*
*et de Développements en Intelligence Artificielle*

# Deconstructing Multi-Objective Evolutionary Algorithms: an Iterative Analysis on the Permutation Flow-Shop Problem

L.C.T. Bezerra, M. López-Ibáñez, and T. Stützle

**IRIDIA – Technical Report Series**

Technical Report No.
TR/IRIDIA/2014-008

March 2014

**IRIDIA – Technical Report Series**
ISSN 1781-3794

Published by:
IRIDIA, *Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle*
Université Libre de Bruxelles
Av F. D. Roosevelt 50, CP 194/6
1050 Bruxelles, Belgium

Technical report number TR/IRIDIA/2014-008

# Deconstructing Multi-objective Evolutionary Algorithms: An Iterative Analysis on the Permutation Flow-Shop Problem

Leonardo C. T. Bezerra, Manuel López-Ibáñez, and Thomas Stützle

IRIDIA, Université Libre de Bruxelles (ULB), Brussels, Belgium
{lteonaci,manuel.lopez-ibanez,stuetzle}@ulb.ac.be

**Abstract.** Many studies in the literature have applied multi-objective evolutionary algorithms (MOEAs) to multi-objective combinatorial optimization problems. Few of them analyze the actual contribution of the basic algorithmic components of MOEAs. These components include the underlying EA structure, the fitness and diversity operators, and their policy for maintaining the population. In this paper, we compare seven MOEAs from the literature on three bi-objective and one tri-objective variants of the permutation flowshop problem. The overall best and worst performing MOEAs are then used for an iterative analysis, where each of the main components of these algorithms is analyzed to determine their contribution to the algorithms' performance. Results confirm some previous knowledge on MOEAs, but also provide new insights. Concretely, some components only work well when simultaneously used. Furthermore, a new best-performing algorithm was discovered for one of the problem variants by replacing the diversity component of the best performing algorithm (NSGA-II) with the diversity component from PAES.

## 1 Introduction

Evolutionary algorithms (EAs) are one of the most widely used metaheuristic algorithms and since a long time attract a large research community. Even when considering only applications to multi-objective optimization problems, many different multi-objective EAs (MOEAs) have been proposed [1, 3, 6, 9, 11, 19, 20]. In fact, MOEAs were among the first metaheuristics applied to multi-objective combinatorial optimization (MCOP) [16]. Moreover, several relevant developments in heuristic algorithms for multi-objective optimization have been advanced in the research efforts targeted to MOEAs. Such developments include archiving [11], dominance-compliant performance measures [21] and the performance assessment of multi-objective optimizers [22].

Despite the large number of MOEAs proposed in the literature, little effort has been put into understanding the actual impact of specific algorithmic components. In general, the efficacy of MOEAs depends on a few main components. The first, common to single-objective optimization, is the underlying EA structure, which includes genetic algorithms (GA), evolutionary strategies (ES) and differential evolution (DE). The other two components, *fitness* and *diversity* operators, have been adapted from single-objective optimization to deal with

search aspects particular to multi-objective problems. In MOEAs, the fitness operator typically considers the Pareto dominance relations between chromosomes in order to intensify the search. Conversely, the diversity operators focus on spreading the solutions over the objective space in order to find a set of trade-off solutions representative of various possible preferences. Finally, the policy for the population management addresses the issue of which individuals to remove after new ones have been generated by the evolutionary operators (one may call this also population reduction policy). These policies make use of fitness and diversity measures, but the frequency with which they are computed may differ from algorithm to algorithm.

Traditionally, a new MOEA is proposed as a monolithic block that integrates specific choices for the fitness, diversity, and population reduction. In this way, it is difficult to understand the actual impact each of these components has on performance. Often, algorithms that differ only by few such components have not been compared directly. In this paper, we compare seven different MOEAs on the permutation flowshop problem (PFSP) to understand their performance. We consider the three most relevant objective functions used in the PFSP literature, namely *makespan*, *total flow time*, and *total tardiness*. We then implement and compare all MOEAs for the three possible bi-objective variants as well as for the tri-objective variant.

While the performance of some algorithms is consistent across all PFSP variants, others present major differences. We then analyze on all variants, iteratively moving from the worst to the best ranked algorithm in the configuration space of MOEAs. This is done in a fashion akin to path relinking [10] by replacing component by component in the worst performing algorithm until obtaining the structure of the best performing one. Such a type of analysis has been proposed recently by Fawcett and Hoos [8] in the context of automatic algorithm configuration. The goal of our analysis is to identify the algorithm components that, for a specific problem, contribute most to algorithm performance. The results of the iterative analysis conducted show that some algorithms can be easily improved by means of additional convergence pressure. Furthermore, for one of the PFSP variants the best-performing algorithm can be improved by replacing its diversity component with the component used by the worst-performing algorithm.

The paper is organized as follows. Section 2 presents the PFSP. Section 3 describes the algorithms we consider in this work, highlighting the differences between them. The experimental setup is presented in Section 4. The comparison of the MOEAs and the results of the analysis of MOEA components is presented in Sections 5 and 6, respectively. Finally, conclusions and possibilities for future work are discussed in Section 7.

## 2 The Permutation Flowshop Problem

The PFSP is one of the most widely studied scheduling problems in operations research. It arises in various industries such as chemical, steel, or ceramic tile production where jobs have to be executed by different machines in a given order. Since each execution takes a different amount of time, the order in which jobs are processed is of major importance for the efficiency of the process. An

instance of the PFSP consists of a set of $n$ jobs and $m$ machines and a matrix $P$ of $n \times m$ processing times $p_{ij}$, where $p_{ij}$ is the processing time of job $i$ on machine $j$. For a permutation $\pi$ that represents the order in which jobs will be executed, the completion times of all jobs on all machines are defined as

$$C_{\pi_0,j} = 0, \quad j = 1, ..., m, \qquad C_{\pi_i,0} = 0, \quad i = 1, ..., n, \tag{1}$$

$$C_{\pi,j} = \max\{C_{\pi_{i-1},j}, C_{\pi_i,j-1}\}, \qquad i = 1, ..., n,\ j = 1, ..., m \tag{2}$$

where $\pi_i$ is the job at position $i$ in the permutation, $\pi_0$ is a dummy job, and machine 0 is a dummy machine. Typically, the PFSP has been studied using various objective functions, the most used being (i) *makespan* ($\mathsf{C}_{\max}$), i.e., the completion time of the last job on the last machine; (ii) *total flow time* ($\mathsf{TFT}$), i.e., the sum of the completion times of each job on the last machine; and (iii) *total tardiness* ($\mathsf{TT}$), the difference between the completion times of all jobs in the last machine and their due dates. In the latter case, a list of due dates is provided, where $d_i$ is the due date of job $i$.

When more than one of these objectives is considered simultaneously, solutions are compared not based on a single objective value, but on an objective *vector*. Given a PFSP variant with objectives $f^i, i = 1, \ldots, k$, a solution $s$ is said to be better (or to *dominate*) another solution $s'$ if $\forall i$, $f^i(s) \leq f^i(s')$ and $\exists\, i$, $f^i(s) < f^i(s')$. If neither solution dominates the other, they are said to be *nondominated*. A typical goal of optimizers designed to solve a multi-objective problem is to find the set of nondominated solutions w.r.t. all feasible solutions, the Pareto set. Since this may prove to be computationally unfeasible, multi-objective metaheuristics have been used to find approximation sets, i.e., sets whose image in the objective space (called *approximation fronts*) best approximate the Pareto set image.

In this paper, we implement the MOEAs to solve the three possible bi-objective variants that combine $\mathsf{C}_{\max}$, $\mathsf{TFT}$, and $\mathsf{TT}$, namely $\mathsf{C}_{\max}$-$\mathsf{TFT}$, $\mathsf{C}_{\max}$-$\mathsf{TT}$, and $\mathsf{TFT}$-$\mathsf{TT}$. Moreover, we also consider the tri-objective variant $\mathsf{C}_{\max}$-$\mathsf{TFT}$-$\mathsf{TT}$. To ensure the algorithms efficacy, we use the same algorithmic components typically found in the PFSP literature. Solutions are represented through direct encoding, that is, each individual in the MOEA is a permutation of the jobs. Initial solutions are generated randomly as traditionally done in MOEAs. The crossover operator applied is the two-point crossover operator. Finally, two mutation operators are considered: *insert*, which selects a job uniformly at random, and reinserts it in a position of the permutation that is also chosen uniformly at random, and *exchange*, which swaps two jobs of the permutation that are also chosen uniformly at random.

## 3 Multi-objective evolutionary algorithms

Since the first proposal of a MOEA [16], several algorithmic structures and components have been devised. To better understand the commonalities and peculiarities of the most relevant approaches, we review various proposals here. All MOEAs described below and used in this work are summarized in Table 1.

**Table 1.** Main algorithmic components of the MOEAs considered in this work. For an explanation of the table entries we refer to the text.

| algorithm | fitness | diversity | reduction | structure |
|---|---|---|---|---|
| MOGA [9] | *dominance rank* | *niche sharing* | *one shot* | *GA* |
| NSGA-II [6] | *dominance depth* | *crowding distance* | *one shot* | *GA* |
| SPEA2 [20] | *dominance strength* | *k-NN* | *iterative* | *GA* |
| IBEA [19] | *binary indicator* | *none* | *iterative* | *GA* |
| HypE [1] | *hypervolume contribution* | *none* | *iterative* | *GA* |
| PAES [11] | *none* | *grid crowding* | *one shot* | $(1+1)$-*ES* |
| SMS-EMOA [3] | *three-way fitness* | *none* | *steady-state* | $(\mu+1)$-*ES* |

Many extensions of EAs to multi-objective optimization rely mostly on the extension of the concepts of fitness and diversity. In a MOEA, the fitness of a solution is generally calculated by means of dominance compliant metrics, meaning the algorithm will favor solutions according to Pareto dominance. Several fitness metrics can be found in the literature, such as dominance rank [9], dominance strength [20], and dominance depth [1, 6]. Besides fitness measures, the population is also evaluated according to diversity metrics. In single-objective optimization, diversity metrics are used to prevent the algorithm from stagnating by spreading individuals across the decision space. This concept becomes even more important for multi-objective optimization since multiple solutions need to be found. In this context, diversity is generally measured in terms of the objective space, the main concern being to have a well-distributed approximation to the Pareto front. The most commonly used metrics include crowding distance [6], niche sharing [17], and k-nearest-neighbor [20]. Finally, algorithms also differ as to the frequency with which these values are calculated. *One shot* algorithms compute fitness and diversity values once before population reduction and then discard the worst individuals. By contrast, *iterative* algorithms re-calculate fitness and diversity values every time a solution is discarded from the population. Although this second alternative is known to be computationally more expensive, initial results have shown this strategy to produce better results when runtime is not an issue [1].

The particular choice of fitness, diversity metrics and how often these are computed are distinguishing features of different multi-objective EAs. The most relevant algorithms propose their own fitness and diversity strategies. MOGA [9], for instance, uses dominance ranking and niche sharing. The population reduction adopts the one shot policy. NSGA-II [6] uses dominance depth and crowding distance, and also uses one shot population reduction. SPEA2 [20] uses a combination of dominance count and dominance rank for the fitness computation and a k-NN metric for diversity, but discards individuals using an iterative reduction policy. IBEA [19] uses binary quality indicators to compare solutions. The two most commonly adopted are: (i) the $\epsilon$-indicator ($I_\epsilon$), that computes the $\epsilon$ value that would have to be added (or multiplied) to one solution for it to be dominated by another, and; (ii) the hypervolume difference ($I_H^-$), which given a pair of solutions computes the volume of the subspace one individual dominates that the other does not. These binary values are computed for each pair of solutions

in the population. The fitness of an individual is then equal to the aggregation of its indicator w.r.t. the rest of the population.

More recently, the hypervolume indicator has been used to evaluate fitness and diversity simultaneously during an MOEA run. In this case, it computes the volume of the objective space dominated by a given approximation set, bounded by a reference point. The hypervolume used as a fitness metric captures both concepts of closeness to the Pareto front and spread of the approximation, thus replacing the explicit diversity measure in other algorithms [1, 19]. For example, HypE [1] is a traditional genetic algorithm (GA) that uses the hypervolume contribution, that is, the volume of the subspace dominated exclusively by a given solution. HypE evaluates the fitness of the individuals at two moments: (i) before mating, when all individuals are assessed, and; (ii) during population reduction, when a speed-up is employed: the hypervolume contribution is used as a tie-breaker for the dominance depth approach.

Several MOEAs are based on the structure of evolution strategies (ES). PAES [11] is a (1+1)-ES that actually resembles a local search procedure. At each iteration, an incumbent solution is mutated and compared to the population of the algorithm, which maintains only nondominated solutions. The actual efficiency of the algorithm lies in the adaptive procedure used to keep this population well-spread and to direct the search towards regions that are little explored. If the population size has not yet reached the maximum allowed size, the new solution is accepted as long as it is not dominated by an existing solution. Otherwise, the new solution is only accepted in the population if it either dominates an existing solution or if it is located in a region of the objective space where the algorithm still has not found many solutions.

Another multi-objective ES proposal is SMS-EMOA [3], a steady-state ES. For mating selection, random individuals are uniformly chosen. This algorithm resembles HypE regarding the population fitness metric, as SMS-EMOA also uses dominance depth followed by hypervolume contribution for tie-breaking. Particularly, the combined fitness metric used by this algorithm can be described as a three-way fitness metric. First, individuals are sorted according to dominance depth. If all individuals in the population are given the same fitness value, the hypervolume contribution is used to break ties. Otherwise, all fronts that fit the new population are preserved, and tie-breaking (by means of the dominance rank metric) is applied for the first front that does not fit fully into the population. As SMS-EMOA is a steady-state algorithm, the offspring always replaces the worst individual of the parent population, and the mating selection is always done at random.

## 4 Experimental setup

We use the benchmark set provided by Taillard [18] following previous work on multi-objective PFSP [7, 15]. This benchmark set contains instances with all combinations of $n \in \{20, 50, 100, 200\}$ jobs and $m \in \{5, 10, 20\}$ machines, except for $n = 200$ and $m = 5$ (200x5). We consider 10 instances of each size, 110 instances in total. The maximum runtime per instance equals $t = 0.1 \cdot n \cdot m$ seconds. All experiments were run on a single core of Intel Xeon E5410 CPUs,

**Table 2.** Parameter space for tuning the MOEA parameters.

| Parameter | *pop* | *off* | $p_C$ | $p_{mut}$ | $p_X$ |
|---|---|---|---|---|---|
| Domain | $\{10, 20, 30$ $50, 80, 100\}$ | 1 or $[0.1, 2]$ | $[0, 1]$ | $[0, 1]$ | $[0, 1]$ |

| Algorithm | IBEA | MOGA | PAES | SPEA2 |
|---|---|---|---|---|
| Parameter | *indicator* | $\sigma_{share}$ | $l$ | $k$ |
| Domain | $\{I_\epsilon, I_H^-\}$ | $[0.1, 1]$ | $\{1, 2\}$ | $\{1, \ldots, 9\}$ |

running at 2.33GHz with 6MB of cache size under Cluster Rocks Linux version 6.0/CentOS 6.3.

The MOEAs used in this algorithm were instantiated using the C++ ParadisEO framework [12]. We implemented several algorithmic components required for our study that were not available in ParadisEO. We also extended ParadisEO's PFSP library to handle all PFSP variants considered in this paper. The original MOEAs were not designed for the PFSP, and, hence, their parameter settings are likely not well suited for this problem. Therefore, we tuned the parameter settings of all MOEAs using irace [2,13] with a tuning budget of 1000 experiments. As training instances during tuning, we used a different benchmark set [7] from the one used in the final analysis. irace was originally designed for single-objective algorithms, but it has been extended to handle the multi-objective case by using the hypervolume quality measure. For computing the hypervolume, we normalize objective values in the range $[1, 2]$ and use $(2.1, 2.1)$ as the reference point. The parameter space considered for all algorithms is the same, depicted in Table 2, where *pop* is the population size, *off* is the number of offspring, which can be either 1 or relative to the population size *pop*; $p_C$ is the crossover probability, and; $p_{mut}$ is the mutation probability used for determining if an individual will undergo mutation or not. If mutation is applied, a random uniform flip selects between the exchange operator (if the random number is below $p_X$) or the insertion operator (else). The additional parameters required by specific algorithms are below the corresponding MOEA. To overcome known archiving issues of some MOEAs, we add an unbounded external archive to all.

To compare the tuned MOEAs, we consider the average hypervolume over 10 runs of each algorithm per instance. We then plot parallel coordinate plots of these results for each problem variant. Concretely, for each variant we produce 11 plots (one per instance size), each depicting the behavior of all MOEAs in ten different instances. Due to space limitations, few representative results are shown here. The complete set of results are made available as a supplementary page [5]. Finally, to select the source and target algorithms for the ablation analysis, we compute rank sums considering the average hypervolume of the 10 runs per instance size.

## 5 Comparison of MOEAs

The four variants of the PFSP considered in this paper differ significantly from each other in the shape of the non-dominated reference fronts and the number of non-dominated points [7]. As a consequence, both the parameters selected by irace for the MOEAs and their performance may vary greatly from one variant to another.

The parameters of the tuned MOEAs are shown in Table 3. Clearly, the parameters of each MOEA vary a lot across the variants, with the exception

**Table 3.** Parameter settings chosen by irace for all MOEAs on Cmax-TFT. Column *Other* refers to parameters specific to a given MOEA.

| **Cmax-TFT** | | | | | | |
|---|---|---|---|---|---|---|
| MOEA | *pop* | *off* | $p_C$ | $p_{mut}$ | $p_X$ | *Other* |
| HypE | 30 | 91% | 29% | 78% | 28% | - |
| IBEA | 30 | 96% | 15% | 91% | 27% | $I_\epsilon$ |
| MOGA | 50 | 160% | 14% | 67% | 36% | $\sigma = 0.39$ |
| NSGA-II | 20 | 123% | 38% | 90% | 37% | - |
| PAES | 10 | - | - | - | 7% | $l = 2$ |
| SMS-EMOA | 10 | - | 31% | 76% | 34% | - |
| SPEA2 | 20 | 128% | 20% | 91% | 31% | $k = 3$ |

| **Cmax-TT** | | | | | | |
|---|---|---|---|---|---|---|
| MOEA | *pop* | *off* | $p_C$ | $p_{mut}$ | $p_X$ | *Other* |
| HypE | 30 | 88% | 68% | 100% | 25% | - |
| IBEA | 30 | 109% | 44% | 97% | 17% | $I_\epsilon$ |
| MOGA | 30 | 121% | 56% | 86% | 33% | $\sigma = 0.32$ |
| NSGA-II | 20 | 104% | 25% | 96% | 7% | - |
| PAES | 10 | - | - | - | 9% | $l = 2$ |
| SMS-EMOA | 10 | - | 14% | 93% | 14% | - |
| SPEA2 | 10 | 128% | 47% | 86% | 38% | $k = 3$ |

| **TFT-TT** | | | | | | |
|---|---|---|---|---|---|---|
| MOEA | *pop* | *off* | $p_C$ | $p_{mut}$ | $p_X$ | *Other* |
| HypE | 100 | 179% | 13% | 100% | 53% | - |
| IBEA | 30 | 157% | 75% | 96% | 18% | $I_\epsilon$ |
| MOGA | 20 | 129% | 23% | 84% | 48% | $\sigma = 38$ |
| NSGA-II | 80 | 162% | 62% | 100% | 41% | - |
| PAES | 30 | - | - | - | 21% | $l = 2$ |
| SMS-EMOA | 10 | - | 31% | 90% | 34% | - |
| SPEA2 | 20 | 49% | 44% | 87% | 33% | $k = 1$ |

| **Cmax-TFT-TT** | | | | | | |
|---|---|---|---|---|---|---|
| MOEA | *pop* | *off* | $p_C$ | $p_{mut}$ | $p_X$ | *Other* |
| HypE | 50 | 158% | 82% | 96% | 29% | - |
| IBEA | 30 | 157% | 51% | 68% | 44% | $I_\epsilon$ |
| MOGA | 20 | 120% | 69% | 96% | 36% | $\sigma = 0.81$ |
| NSGA-II | 50 | 171% | 25% | 79% | 32% | - |
| PAES | 10 | - | - | - | 19% | $l = 2$ |
| SMS-EMOA | 10 | - | 41% | 87% | 40% | - |
| SPEA2 | 10 | 142% | 53% | 57% | 31% | $k = 6$ |

**Table 4.** Rank sum analysis: The MOEAs are sorted according to their sum of ranks (in parenthesis) for each MO-PFSP variant. Lower rank-sums indicate better performance.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Cmax-TFT** | NSGA-II (218) | IBEA (365) | HypE (372) | SPEA2 (427) | MOGA (477) | SMS-EMOA (508) | PAES (713) |
| **Cmax-TT** | NSGA-II (284) | HypE (316) | SPEA2 (350) | IBEA (386) | MOGA (411) | SMS-EMOA (602) | PAES (731) |
| **TFT-TT** | NSGA-II (252) | MOGA (348) | IBEA (400) | SPEA2 (407) | SMS-EMOA (469) | HypE (537) | PAES (664) |
| **Cmax-TFT-TT** | NSGA-II (176) | SPEA2 (318) | MOGA (339) | IBEA (400) | SMS-EMOA (584) | HypE (583) | PAES (680) |

of population size, which is often small. Using small population sizes is advantageous to MOEAs in the presence of unbounded external archive for two main reasons. First, by using a small population size the algorithm is able to increase its convergence pressure. The second reason is related to minimizing computational overheads. Traditionally, the highest computational costs during one generation in a MOEA are due to (i) function evaluations, (ii) dominance comparisons, and (iii) fitness/diversity metrics computation both in mating and population reduction. The latter can be minimized by a reduced population size and a higher number of offspring produced at each iteration. This way, the number of generations (and hence of fitness/diversity computations) is reduced, and the algorithm is able to explore more potential solutions.

Next, we computed for each algorithm its rank sum. In particular, we ranked the average hypervolume of each algorithm on each instance from one (best) to seven (worst) and then summed the ranks of each algorithm across all 110 instances. The rank sums for all variants are given in Table 4. Two commonalities can be identified across the variants: NSGA-II always presents the lowest rank sums, whereas PAES always ranks worst. Given the heterogeneous nature of the results, we then proceed to further discussion, one variant at a time.

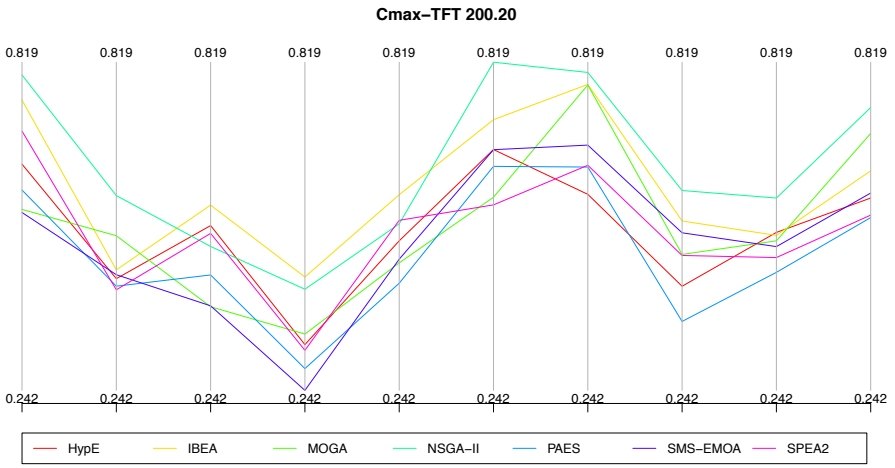


**Fig. 1.** Parallel coordinates plot of the average hypervolume for 10 instances of size 200x20 of Cmax-TFT. Each point on the x-axis represents the average hypervolume obtained over 10 runs on a single instance.

### 5.1 Cmax-TFT

The parameters found by irace for each MOEA for Cmax-TFT (Table 3) follow a pattern. As previously discussed, the population sizes are usually small and the number of offspring is never smaller than 90% of *pop*, but is often higher than 100%. Finally, the mutation rate $p_{mut}$ is always very high and the insertion operator is used much more frequently than the exchange one.

Figure 1 gives parallel coordinate plots for the average hypervolume (given on the $y$-axis) measured on each of the 10 instances of size $n = 200$ and $m = 20$. Clearly, the lines representing the performance of the MOEAs intertwine several times, which shows variability across different instances. Nevertheless, the results are consistent with the ranks depicted in Table 4: NSGA-II and IBEA are always among the best performers, whereas PAES and SMS-EMOA are always among the worst ones. Furthermore, the rank sum difference between IBEA and HypE is too small for statistical significance, as well as the difference between MOGA and SMS-EMOA.

### 5.2 Cmax-TT

The parameters selected by irace for the Cmax-TT variant (Table 3) do not differ much from those obtained above for the Cmax-TFT variant. If anything, the tendencies observed for Cmax-TFT are reinforced: population sizes are smaller, mutation probabilty now almost equals 100% and the exchange mutation operator is used even less often.

When assessing the performances via parallel coordinate plots, though, it is clear in Fig. 2 that the results listed on Table 4 do not match exactly the performance of the MOEAs in all instances. Particularly, for the largest instances

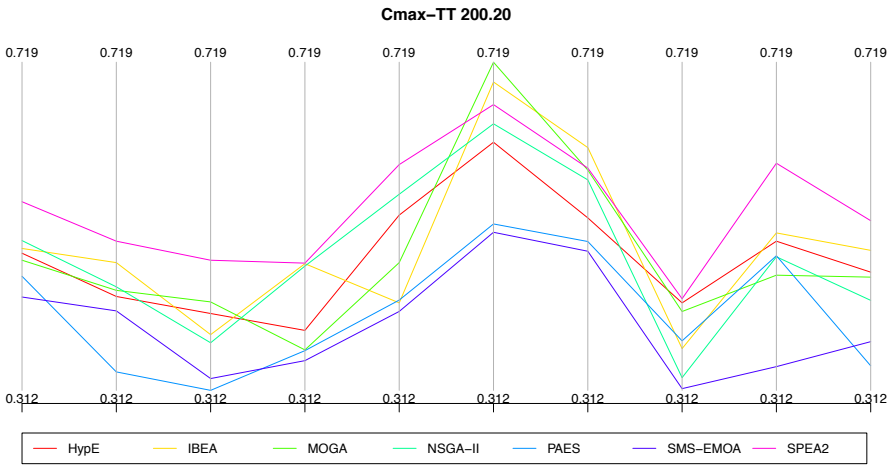


**Fig. 2.** Parallel coordinates plot of the average hypervolume for 10 instances of size 200x20 of Cmax-TT. Each point on the x-axis represents the average hypervolume obtained over 10 runs on a single instance.

it is clear that the MOEA with the best performance this time is SPEA2. Once again, PAES and SMS-EMOA perform rather poorly. The fact that NSGA-II presents a better rank sum than SPEA2 is due to its performance across the whole benchmark.

### 5.3 TFT-TT

Compared to the parameters used for the two previous PFSP variants, the configurations tuned for TFT-TT present two features worth highlighting. First, the frequency of usage of the exchange operator has generally increased. Second, the population maintained by the algorithms became much larger, and with the exception of SPEA2, so has the number of offspring created per generation. These changes are probably due to the variation operators adopted in this work. Although these are commonly used in the state-of-the-art of the PFSP, they have been proposed for optimizing Cmax andmay lose efficiency for TFT or TT.

Concerning solution quality, Fig. 3 shows that NSGA-II performs well for this variant. MOGA also performs well, and again PAES and SMS-EMOA are unable to generate results to match the other MOEAs. When we consider the whole benchmark, the algorithm that loses performance the most is HypE, although it is able to perform almost as many function evaluations as NSGA-II.

### 5.4 Cmax-TFT-TT

The variant Cmax-TFT-TT is the only one comprising three objectives. It is expected that the number of non-dominated solutions is much larger than in the bi-objective variants. A practical consequence of this problem's characteristic

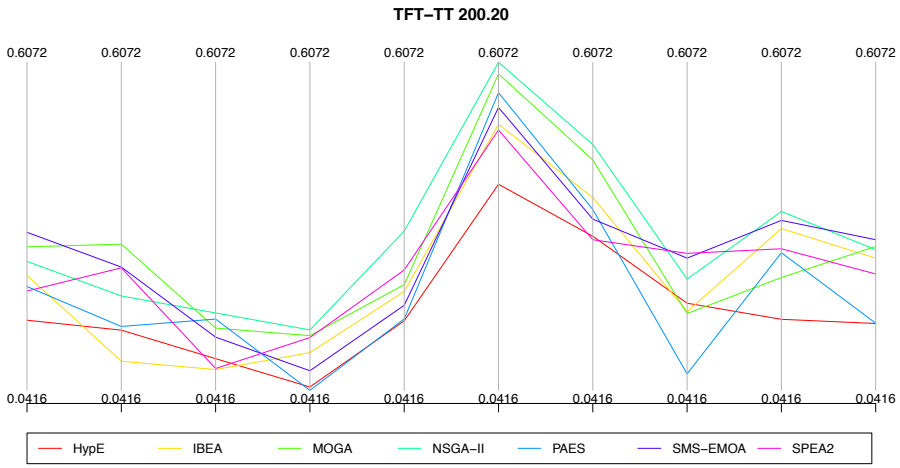


**Fig. 3.** Parallel coordinates plot of the average hypervolume for 10 instances of size 200x20 of TFT-TT. Each point on the x-axis represents the average hypervolume obtained over 10 runs on a single instance.

is that the overhead caused by updating the external archives could slow the algorithms down, allowing a smaller number of function evaluations. This way, algorithms that converge faster to good solutions are likely to be favored, unless they stagnate.

The analysis of the MOEAs' performance on Fig. 4 shows that the tested algorithms can be split in two distinct groups: (i) the best-performing ones, comprised by NSGA-II, IBEA, and SPEA2, and; (ii) the worst-performing ones, with PAES, HypE and SMS-EMOA. The rank sum analysis depicted in Table 4 confirm the most of these observations, except for IBEA. Interestingly, the hypervolume-based algorithms (HypE and SMS-EMOA) do not perform well, even given their different underlying EA structure.

Among all algorithms, the biggest change observed is the low rank sum obtained by NSGA-II. Given that 110 instances are considered for the rank sum analysis, a rank sum of 176 for NSGA-II means that it was consistently the or among the top-ranking algorithms. It is then clear that NSGA-II is a good choice for a practitioner wanting to develop an application for the Cmax-TFT-TT.

## 6 Iterative analysis

The experiments in the previous section showed important differences in MOEA performance in dependence of particular problems. In this section, we conduct an iterative analysis to understand which algorithm components cause the main differences between the best and worst performing MOEA variants, respectively referred to as target and source. This analysis can be seen as a path relinking in the configuration space and it has been applied in the context of automatic algorithm configuration before by Fawcett and Hoos [8]. The main motivation

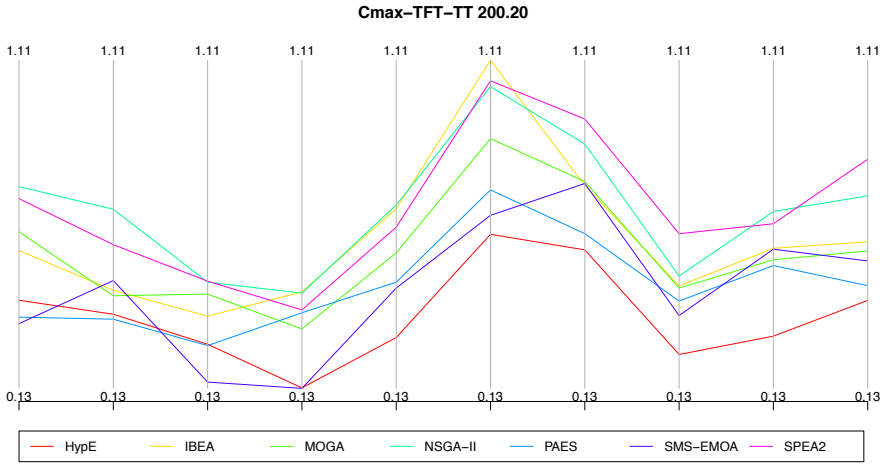


**Fig. 4.** Parallel coordinates plot of the average hypervolume for 10 instances of size 200x20 of Cmax-TFT-TT. Each point on the x-axis represents the average hypervolume obtained over 10 runs on a single instance.

for this analysis is to get insight into the contribution of specific components on algorithm performance. We do so by generating intermediate configurations between the two algorithms. At each step, we modify all individual algorithm components in which the two algorithms differ, and follow the path that has the maximum impact on performance. In this way, the analysis of the intermediate configurations allows us to understand the actual contribution of the individual components to the performance of the algorithm.

We conduct this iterative analysis on the four variants of the PFSP investigated in this work. As source and target algorithms we respectively use PAES and NSGA-II, the worst and best algorithms according to the rank sum analysis for all variants. These algorithms differ in all components considered, providing a rich set of intermediate configurations. Since the EA structure of PAES and NSGA-II is very different, some clarifications are required. First, PAES does not keep an internal population, but a bounded internal archive that can accept a maximum of $|pop|$ solutions. Therefore, a population reduction policy does not make sense in PAES since a single solution is added at a time. Moreover, since only one solution is considered for variation, the crossover operator can never be applied. In this analysis, when switching from the structure used by PAES to the structure used by NSGA-II, all these subcomponents are changed atomically, namely (i) using a population instead of a bounded internal archive; (ii) producing $off$ (or $off \cdot pop$) individuals per generation, (iii) mating selection via binary deterministic tournament (as in NSGA-II), and (iv) using the crossover operator. Moreover, at this point the crossover and mutation probabilities selected by irace for NSGA-II are used, since PAES did not originally present these parameters.

Apart from the underlying EA structural differences between the source and target algorithms, we also consider the numerical parameters selected by irace as

a factor that could interfere with the performance of the algorithms. However, since number of offspring, and crossover and mutation probabilities have already been considered as part of the underlying EA structure component, this factor comprises only the population size and the exchange mutation operator rate (and, consequently, the insertion mutation operator rate).

The analysis conducted showed similar results for Cmax-TFT and Cmax-TFT-TT, but very different results for the other problems. We hence group the discussion of the first two variants, and then individually analyze the other two.

### 6.1 Cmax-TFT and Cmax-TFT-TT

All intermediate configurations tested in the analysis of Cmax-TFT and Cmax-TFT-TT are shown in Fig. 5 (top row). The y-axis represents the rank sums. The x-axis contains the steps of the procedure. In step 0, only the source algorithm is depicted, in this case PAES. In step 1, we modify the four components that differ between PAES and NSGA-II, as previously explained, thus generating four new algorithms.

As shown in Fig. 5 (top row), the component that leads to the strongest decrease of the rank sum is the fitness component. This is a rather intuitive result, given that PAES does not have any component to enforce convergence other than the dominance acceptance criterion of its internal archive. In step 2, we have PAES using the fitness component from NSGA-II, and we test changing its diversity component, its structure, and its numerical parameters, This time, the underlying EA structure becomes the most important factor. At this point, this result is rather expected since it is the component that represents the greater change in the algorithm. Moreover, it also means the diversity component of PAES is indeed effective when combined with dominance depth (or, more generally, with a fitness component). Step 3 modifies the diversity and numerical components, starting from the best algorithm in step 2. Modifying the diversity component now leads to a much better rank sum, almost matching the target algorithm.

### 6.2 Cmax-TT

All intermediate configurations tested for this variant are shown in Fig. 5 (bottom left). Again, the best-improving changes in steps 1 and 2 are, respectively, the addition of a fitness component to PAES and changing its structure to a traditional GA. At step 3, surprisingly, none of the intermediate configurations achieve a better rank sum than the configuration selected in step 2. This means that, for this variant, one needs to change both the diversity metric and the numerical parameters at once to reach the final performance of NSGA-II. This could likely be explained by the additional parameter used by PAES that regulates the size of the its grid cells: their sizes may suit the original internal archive/population size of PAES, but not the one from NSGA-II.

### 6.3 TFT-TT

Finally, all intermediate configurations tested for the TFT-TT variant are shown in Fig. 5 (bottom right). As for the previous variants, once again the best-

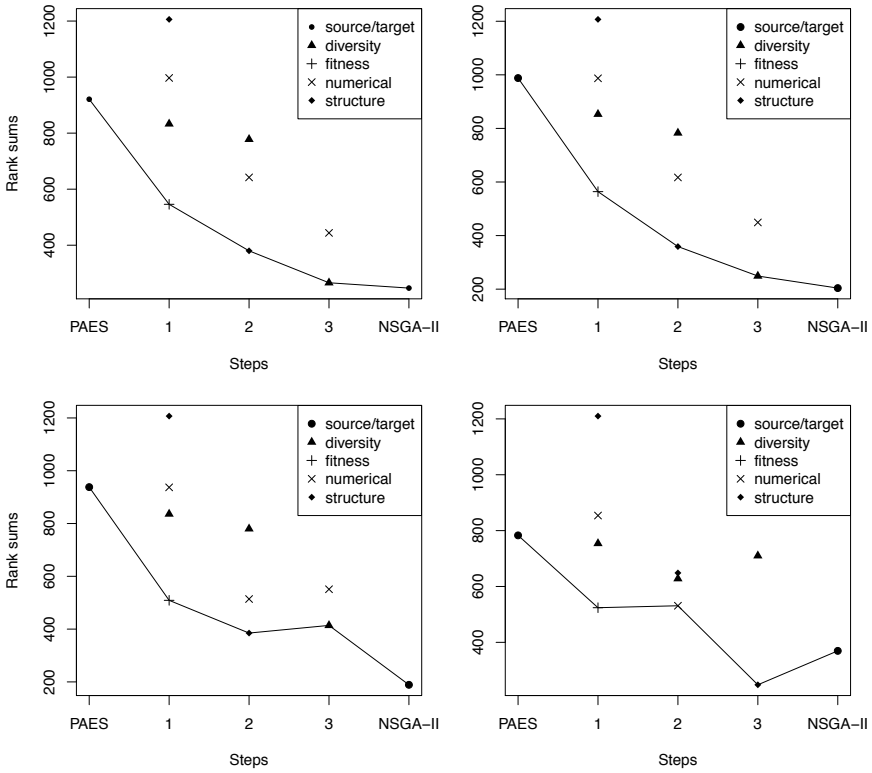


**Fig. 5.** Intermediate configurations tested for Cmax-TFT (top left), Cmax-TFT-TT (top right), Cmax-TT (bottom left), and TFT-TT (bottom right). The line connects the changes that caused the largest performance improvement.

improving change in step 1 is the addition of a fitness component. At step 2, no improvements can be devised, although a change in the numerical parameters does not affect the algorithm significantly. Most interestingly, at step 3 the configuration obtained by changing the structure of the best configuration from step 2 achieves rank sums even lower than those obtained by the target algorithm, NSGA-II. The only difference between the algorithm at step 3 and NSGA-II is the diversity measure. This means that, by simply replacing the diversity metric used by NSGA-II with the one used by PAES, one can devise a better performing algorithm for this problem variant. It is also interesting that this variant was the only one where such a better performing configuration was found, thus reinforcing the idea that it presents peculiar characteristics.

## 7 Conclusions and Future work

Traditionally, MOEAs have been seen as monolithic blocks, which is reflected by the fact that many of these algorithms have been proposed and analyzed as

such. However, for a more detailed analysis it is often preferable to decompose the algorithms into building blocks or, say, main algorithm components. In this paper, we have followed this direction and deconstructed MOEAs into four main components. These components are the underlying EA algorithm, the fitness and diversity operators, and the population management policy. We believe that analyzing these components and their contributions to performance is key to understanding which MOEA works best on each problem and to develop better MOEAs in the future.

In this work, we deconstructed seven relevant MOEAs, namely HypE, IBEA, MOGA, NSGA-II, PAES, SMS-EMOA, and SPEA2. We compared them on three bi-objective and one tri-objective variants of the permutation flowshop problem (PFSP). The results are strongly variant and also problem dependent, highlighting particular strengths and weaknesses of each MOEA. Overall, NSGA-II is always able to find good approximation sets. Maybe surprisingly, some algorithms such as HypE are sometimes among the best for some problem variant while they perform poorly for other problem variants. Furthermore, we conduct an iterative analysis interpolating between the best and worst performing algorithms for all PFSP variants. We show that, sometimes, algorithms can be easily improved by changing a single component, leading to a significant improvement in their performance.

The conclusions drawn from this work confirm the great performance variability metaheuristics generally present for combinatorial optimization problems. As said, the best and worst performing MOEAs can be easily improved by replacing single components. This fact also motivates the need for a flexible, component-wise implementation of MOEAs, as well as the specialization of MOEAs to specific problems through automatic algorithm configuration. Initial results on flexible, configurable frameworks for other multi-objective search techniques have shown that this is a very promising path for research [4, 14].

*Acknowledgments.* The research leading to the results presented in this paper has received funding from the Meta-X ARC project, the COMEX project within the Interuniversity Attraction Poles Programme of the Belgian Science Policy Office, and the FRFC project "*Méthodes de recherche hybrides pour la résolution de problèmes complexes*". Leonardo C. T. Bezerra, Manuel López-Ibáñez and Thomas Stützle acknowledge support from the Belgian F.R.S.-FNRS, of which they are a FRIA doctoral fellow, a postdoctoral researcher and a senior research associate, respectively.

## References

1. Bader, J., Zitzler, E.: HypE: An algorithm for fast hypervolume-based many-objective optimization. Evolutionary Computation 19(1), 45–76 (2011)
2. Balaprakash, P., Birattari, M., Stützle, T.: Improvement strategies for the F-race algorithm: Sampling design and iterative refinement. In: Bartz-Beielstein, T., et al. (eds.) Hybrid Metaheuristics, LNCS, vol. 4771, pp. 108–122. Springer (2007)
3. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: Multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. 181(3), 1653–1669 (2007)
4. Bezerra, L.C.T., López-Ibáñez, M., Stützle, T.: Automatic generation of multi-objective ACO algorithms for the biobjective knapsack problem. In: Dorigo, M., et al. (eds.) ANTS 2012, LNCS, vol. 7461, pp. 37–48. Springer (2012)

5. Bezerra, L.C.T., López-Ibáñez, M., Stützle, T.: Deconstructing multi-objective evolutionary algorithms: An iterative analysis on the permutation flowshop: Supplementary material. http://iridia.ulb.ac.be/supp/IridiaSupp2013-010/ (2013)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6(2), 181–197 (2002)
7. Dubois-Lacoste, J., López-Ibáñez, M., Stützle, T.: A hybrid TP+PLS algorithm for bi-objective flow-shop scheduling problems. Comput. Oper. Res. 38(8), 1219–1236 (2011)
8. Fawcett, C., Hoos, H.H.: Analysing differences between algorithm configurations through ablation. In: Proceedings of MIC 2013, the 10th Metaheuristics International Conference. pp. 123–132 (2013)
9. Fonseca, C.M., Fleming, P.J.: Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: Forrest, S. (ed.) ICGA. pp. 416–423. Morgan Kaufmann Publishers (1993)
10. Glover, F.: A template for scatter search and path relinking. In: Hao, J.K., et al. (eds.) Artificial Evolution, LNCS, vol. 1363, pp. 1–51. Springer (1998)
11. Knowles, J.D., Corne, D.: Approximating the nondominated front using the Pareto archived evolution strategy. Evolutionary Computation 8(2), 149–172 (2000)
12. Liefooghe, A., Jourdan, L., Talbi, E.G.: A software framework based on a conceptual unified model for evolutionary multiobjective optimization: ParadisEO-MOEO. Eur. J. Oper. Res. 209(2), 104–112 (2011)
13. López-Ibáñez, M., Dubois-Lacoste, J., Stützle, T., Birattari, M.: The irace package, iterated race for automatic algorithm configuration. Tech. Rep. TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium (2011)
14. López-Ibáñez, M., Stützle, T.: The automatic design of multi-objective ant colony optimization algorithms. IEEE Trans. Evol. Comput. 16(6), 861–875 (2012)
15. Minella, G., Ruiz, R., Ciavotta, M.: A review and evaluation of multiobjective algorithms for the flowshop scheduling problem. INFORMS Journal on Computing 20(3), 451–471 (2008)
16. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms. In: Grefenstette, J.J. (ed.) ICGA-85. pp. 93–100. Lawrence Erlbaum Associates (1985)
17. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2(3), 221–248 (1994)
18. Taillard, É.D.: Benchmarks for basic scheduling problems. Eur. J. Oper. Res. 64(2), 278–285 (1993)
19. Zitzler, E., Künzli, S.: Indicator-based selection in multiobjective search. In: Yao, X., et al. (eds.) Parallel Problem Solving from Nature, PPSN VIII, LNCS, vol. 3242, pp. 832–842. Springer (2004)
20. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In: Giannakoglou, K.C., et al. (eds.) Evolutionary Methods for Design, Optimisation and Control. pp. 95–100. CIMNE, Barcelona, Spain (2002)
21. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto evolutionary algorithm. IEEE Trans. Evol. Comput. 3(4), 257–271 (1999)
22. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Grunert da Fonseca, V.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. 7(2), 117–132 (2003)