Neutral but a Winner! How Neutrality Helps Multiobjective Local Search Algorithms

Aymeric Blot[1,2], Hernán Aguirre[4], Clarisse Dhaenens[2,3], Laetitia Jourdan[2,3], Marie-Eléonore Marmion[2,3], and Kiyoshi Tanaka[4]

[1] ENS Rennes, Ker Lann; Université Rennes 1, France
[2] Inria Lille - Nord Europe, DOLPHIN Project-team, France
[3] Université Lille 1, LIFL, UMR CNRS 8022, France
[4] Faculty of Engineering, Shinshu University, Nagano, Japan
aymeric.blot@ens-rennes.fr,
{clarisse.dhaenens;laetitia.jourdan;marie-eleonore.marmion}@lifl.fr,
{ahernan;ktanaka}@shinshu-u.ac.jp

Abstract. This work extends the concept of neutrality used in single-objective optimization to the multi-objective context and investigates its effects on the performance of multi-objective dominance-based local search methods. We discuss neutrality in single-objective optimization and fitness assignment in multi-objective algorithms to provide a general definition for neutrality applicable to multi-objective landscapes. We also put forward a definition of neutrality when Pareto dominance is used to compute fitness of solutions. Then, we focus on dedicated local search approaches that have shown good results in multi-objective combinatorial optimization. In such methods, particular attention is paid to the set of solutions selected for exploration, the way the neighborhood is explored, and how the candidate set to update the archive is defined. We investigate the last two of these three important steps from the perspective of neutrality in multi-objective landscapes, propose new strategies that take into account neutrality, and show that exploiting neutrality allows to improve the performance of dominance-based local search methods on bi-objective permutation flowshop scheduling problems.

Keywords: neutrality, multi-objective optimization, local search, permutation flowshop scheduling

1 Introduction

In the single-objective context, solving large optimization problems with local search approaches allows to obtain good solutions in a reasonable time [6]. These local search methods are based on a neighborhood relation that enables to perform local improvements. It has been shown that such methods are sensitive to the properties of the landscape of the problem studied, and that it is crucial to analyze and understand such properties in order to improve the performance of the algorithms.

This work focuses on neutrality, a property that characterizes neighboring solutions having the same fitness. In single objective-optimization it is known that the degree of neutrality of a landscape impacts the behavior of local search methods. There are also several studies showing that exploiting neutrality in a local search method can improve performance of the method [7].

In the multi-objective context, there are also efficient local search methods that have been proposed to approximate the Pareto optimal set, such as the Dominance based Multi-objective Local Search (DMLS) [4]. However, not much is known about neutrality, its effects, and how to take advantage of it in order to improve the performance of multi-objective algorithms. Indeed, the performance of a DMLS algorithm is closely related to the geometry of the landscape of the problem to solve. Moreover, the Pareto dominance relation induces landscapes where many solutions cannot be compared with many others (solutions equivalent in term of quality), and one major difficulty of DMLS algorithms is, at each iteration, to choose a selected neighbor which may be *equivalent*, i.e. incomparable with the current explored solution.

This work extends the concept of neutrality to multi-objective optimization with the aim to analyze whether exploiting neutrality is also beneficial in a multi-objective context. We discuss neutrality in single-objective optimization and fitness assignment in multi-objective algorithms to provide a general definition for neutrality applicable to multi-objective landscapes. We also put forward a definition of neutrality when Pareto dominance is used to compute fitness of solutions. Then, we analyze existing DMLS from the point of view of neutrality, in order to propose new efficient schemes. We focus on strategies that take into account neutrality, particularly during the neighborhood exploration and the creation of the candidate set of solutions to update the archive, showing that exploiting neutrality allows to improve the performance of dominance-based local search methods on bi-objective permutation flowshop scheduling problems.

The paper is organized as follows. Section 2 states background definitions of multi-objective combinatorial optimization. It presents the problem that will be used as an illustration, the Permutation Flowshop Scheduling Problem (PFSP) and Dominance based Multi-objective Local Search approaches (DMLS). In Section 3, we propose a multi-objective concept of neutrality, and analyze its integration in existing DMLS algorithms. This leads us to propose several improvements to DMLS algorithms to efficiently incorporate this notion. In Section 4, experiments are conducted in order to emphasize the importance of taking care of neutrality in DMLS algorithms and to measure the impact of our propositions on the Permutation Flowshop Scheduling Problem (PFSP). At last, Section 5 gives the conclusions of the presented work and future research interests.

2 Background

This work investigates the effects of neutrality focusing on Dominance-based Local Search Methods using the Bi-objective Permutation Flowshop Scheduling Problem as an illustrative example. This section describes the optimization problem and the local search methods used, together with necessary notation to better understand the paper.

2.1 The Bi-objective Permutation Flowshop Scheduling Problem

The Permutation Flowshop Scheduling Problem (PFSP) is a multi-objective combinatorial optimization (MoCO) problem widely investigated in the literature. The PFSP consists in scheduling a set of N jobs $\{J_1, \ldots, J_N\}$, on a set of M machines $\{M_1, \ldots, M_M\}$. Machines are critical resources that can only process one job at a time. A job J_i is composed of M tasks $\{t_{i,1}, \ldots, t_{i,M}\}$ for the M machines respectively. A processing time $p_{i,j}$ is associated to each task $t_{i,j}$, and a due date d_i is associated with each job J_i. The operating sequence is the same on every machine. Therefore, a schedule may be represented as a permutation of jobs $\pi = \{\pi_1, \ldots, \pi_N\}$. Ω is the set of the feasible solutions.

The two objectives, f_1 and f_2, considered in this paper are the makespan $C_{\max}$ (eq. 1), i.e. the total completion time, and the total tardiness T (eq. 2). Both objectives have to be minimized.

$$f_1 = C_{\max} = \max_{i \in \{1,\ldots,N\}} \{C_{\pi_i}\} \tag{1}$$

$$f_2 = T = \sum_{i=1}^{N} \{ \max \; \{0, \; C_{\pi_i} - d_{\pi_i}\}\} \tag{2}$$

The feasible outcome vectors of the *objective space* are compared using *Pareto dominance* $\succ$. In this minimization context, a solution $x \in \Omega$ is said to dominate a solution $y \in \Omega$, denoted by $x \succ y$, if they satisfy relation (3).

$$\forall i \in \{1, 2\}, \; f_i(x) \leq f_i(y) \; \bigwedge \; \exists i \in \{1, 2\}, \; f_i(x) < f_i(y) \tag{3}$$

If solution y is non-dominated by solution x we denote $y \nprec x$.

This paper focuses on multi-objective local search methods based on a neighborhood definition. The neighborhood considered in this paper uses the insertion operator, where a job located at position i is inserted at position $j \neq i$ and the jobs located between positions i and j are shifted. The number of neighbors per solution is then $(N-1)^2$.

2.2 Dominance-based Multi-objective Local Search

In the literature, numerous methods have been proposed to solve MoCO problems. The Dominance-based Multi-objective Local Search algorithms represent a class of local search approaches designed to approximate the Pareto front of a MoCO [4] problem. DMLS algorithms keep an archive of mutually non-dominated solutions and uses a neighborhood structure to improve the solutions of the archive. The main steps of a DMLS algorithm are as follows.

Step 1 Initialize the archive with a randomly created solution x, $\mathcal{A} \leftarrow \{x\}$.

Step 2 Select from the archive a set of solutions for exploration, $\mathcal{X} \subseteq \mathcal{A}$.

Step 3 For each solution $x \in \mathcal{X}$, explore the neighborhood of x until a neighbor z fulfilling a required criterion is found. During exploration of x, in addition to z, neighbor solutions x' non-dominated by x are collected as candidate solutions to update the archive, $\mathcal{C} = \{x' \in \mathcal{N}(x) \mid x' \not\prec x\} \cup \{z\}$.

Step 4 Update the archive $\mathcal{A}$ with the collected candidate solutions $\mathcal{C}$ making sure that only non-dominated solutions remain in the archive.

Step 5 If termination criterion is not met, repeat from Step 2.

Step 6 Return the archive $\mathcal{A}$.

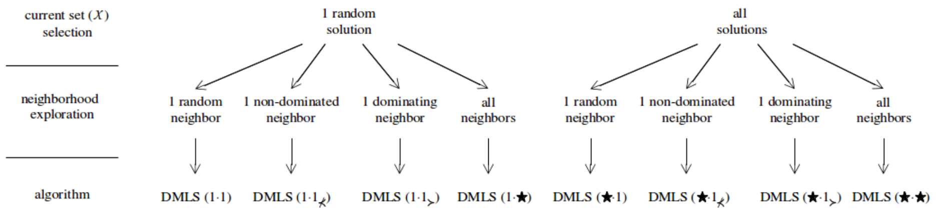

Fig. 1. Nomenclature of DMLS algorithms.

Several strategies are defined for Step 2 and Step 3, which lead to different configurations of DMLS algorithms. A specific nomenclature and classification of DMLS algorithms was proposed by Liefooghe et al. [4], as shown in Figure 1. In Step 2, the candidate set for exploration $\mathcal{X}$ can be obtained by selecting from the archive either *one* solution randomly (DMLS $(1 \cdot \quad)$) or *all* solutions (DMLS $(\bigstar \cdot \quad)$). In Step 3, the neighborhood exploration of each solution $x \in \mathcal{X}$ can be either *exhaustive* or *partial*. If it is exhaustive (DMLS $(\quad \cdot \bigstar)$), all the neighbors are visited and all non-dominated neighbors $x' \not\prec x$ are collected in the candidate set $\mathcal{C}$ to update the archive. If the exploration is partial, different strategies can be used. A possible partial exploration strategy is to accept a random neighbor whatever its quality (DMLS $(\quad \cdot 1)$). This strategy corresponds to a random search. Other strategy is to explore the neighborhood of a solution until a dominating neighbor $z \succ x$ is found (DMLS $(\quad \cdot 1_{\succ})$). A third partial exploration strategy is to explore the neighborhood of a solution until a non-dominated solution is found $z \not\prec x$ (DMLS $(\quad \cdot 1_{\not\prec})$), in which case z could be a dominating solution $z \succ x$ or mutually non-dominated with respect to x, $z \not\prec x$ `and` $x \not\prec z$.

Liefooghe et al. [4] experimented on the bi-objective PFSP showing that some DMLS configurations perform better than others. In the rest of the paper, only the following configurations DMLS $(1 \cdot 1_{\not\prec})$, DMLS $(1 \cdot 1_{\succ})$ and DMLS $(\bigstar \cdot 1_{\succ})$ are considered.

3 Neutrality extended to Multi-Objective Optimization

In this section, we discuss the concept of neutrality in single objective optimization, propose a definition of neutrality in the multi-objective context, particularly for Pareto dominance based approaches, clarify how neutrality has been used so far in the DMLS algorithm, and propose new strategies for DMLS aiming to further exploit neutrality in multi-objective optimization.

3.1 Neutrality in Single-Objective Optimization

In single-objective optimization, neutrality arises when neighboring solutions have the same quality. More formally, let us denote Ω the space of the admissible solutions, $\mathcal{N}$ a neighborhood structure, and f a fitness function. A fitness landscape of the problem is defined by the triplet $(\Omega, \mathcal{N}, f)$. A neutral neighbor of a solution $s \in \Omega$ is a neighbor solution $s' \in \Omega$ with the same fitness value, $f(s) = f(s')$. Given a solution $s \in \Omega$, its set of neutral neighbors $N_n(s)$ is defined by:

$$\mathcal{N}_n(s) = \{s' \in \mathcal{N}(s) \mid f(s') = f(s)\}$$

The neutral degree of a solution is the number of its neutral neighbors $|\mathcal{N}_n(s)|$. A fitness landscape is said to be neutral if there are many solutions with a high neutral degree $|\mathcal{N}_n(s)|$.

Neutral solutions can be considered in the design of local search algorithms [1, 7, 10] either to escape from a local optimum or to explore more widely the search space. Since equivalent solutions have proved to be useful in single-objective optimization, we propose to study the effects of exploiting equivalent solutions in multi-objective optimization.

3.2 Neutrality in Multi-Objective Optimization

The definition of neutrality in single-objective optimization is based on neighbor solutions that have same fitness values. In order to give a definition of neutrality in multi-objective optimization, we need to characterize neutral neighbors in this context. Particularly, what means equal fitness of two solutions.

In multi-objective optimization there are various approaches to compute fitness of solutions. These include Pareto dominance, Pareto dominance and density estimation, scalarization functions, and indicator functions such as the hypervolume I_{HV} or the epsilon indicator I_{ϵ^+}. In general, we can say that fitness f is a function of the n single-objective fitness values $f_1, f_2, \cdots, f_n$ computed for a solution, i.e. $f = g(f_1, f_2, \cdots, f_n)$. Thus, a similar definition used for neutrality in single-objective optimization can be used for neutrality in multi-objective optimization. Namely, a multi-objective neutral neighbor of a solution $s \in \Omega$ is a neighbor solution $s' \in \Omega$ with the same fitness value $f(s) = f(s')$, where $g(f_1, f_2, \cdots, f_n)$ is a function of the single-objective fitness values and $f = g(f_1, f_2, \cdots, f_n)$.

It should be noted that each approach to compute fitness in multi-objective optimization implies a different fitness function and therefore a different landscape. This also means that the set of neutral neighbors of a solution might vary depending on the approach used to compute fitness. However, all approaches aim to find the Pareto optimal set of the problem or a good approximation of it. It will be very interesting to study the effects of neutrality in the different approaches to multi-objective optimization. In this work, we restrict our attention to approaches that use Pareto dominance to determine fitness of solutions.

Given a a solution x to explore based on a neighborhood structure $\mathcal{N}$, Pareto dominance implies three types of neighbors x' respect to x: dominating neighbors $x' \succ x$, dominated neighbors $x \succ x'$, or mutually non-dominated neighbors $x \nprec x'$ `and` $x' \nprec x$, as shown in Figure 2. These latter neighbors are non-comparable solutions. Therefore, they can be viewed as *equivalent* neighbors, or same fitness neighbors, that define the neutral neighbors in the multi-objective context when fitness of solutions is computed using Pareto dominance. More precisely, given a solution $s \in \Omega$, its set of neutral neighbors is defined by:

$$\mathcal{N}_e(s) = \{s' \in \mathcal{N}(s) \mid s \nprec s' \texttt{ and } s' \nprec s\}$$

Note that this definition includes the case where two neighbors have the same objective vector $(s' \in \mathcal{N}(s), \forall i \in [1, n], f_i(s) = f_i(s'))$.

The motivation to extend neutrality from single- to multi-objective optimization comes from the fact that single-objective local search algorithms can benefit from equivalent/neutral solutions. These solutions allow to continue the search when it is trapped in a local optimum without degrading. Similarly, in multi-objective optimization that uses Pareto dominance to establish fitness of the individuals, a local search algorithm can be trapped in a Pareto local optimum (PLO) and some equivalent/neutral neighbors can help to escape from it. In the following, we use the term *neutral* to qualify these equivalent neighbors.

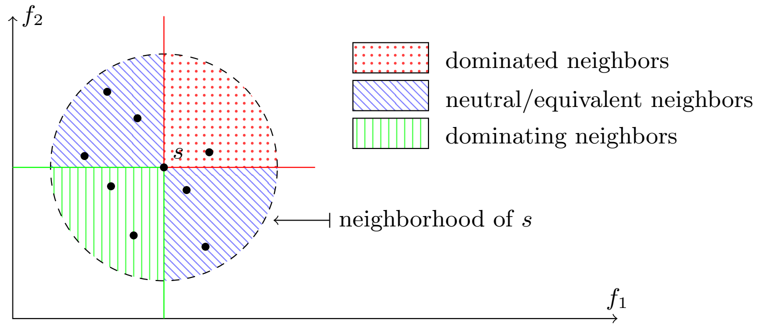

Fig. 2. Neighborhood in bi-objective optimization.

3.3 Neutrality in DMLS Algorithms

Section 2.2 briefly described the DMLS algorithms for multi-objective optimization. Analyzing these algorithms, we can see that some of them can exploit neutral neighbors to approach the Pareto front, but require that the neighbors survive several steps of the algorithm. For example, during Step 3 DMLS $(1 \cdot 1_{\not\succ})$ and DMLS $(\bigstar \cdot 1_{\not\succ})$ algorithms can generate at most one neutral neighbor solutions per $x \in \mathcal{X}$ if and only if during exploration a dominating solution is not found first. On the other hand, DMLS $(1 \cdot 1_{\succ})$ and DMLS $(\bigstar \cdot 1_{\succ})$ can generate more than one neutral neighbor solution per $x \in \mathcal{X}$ until the first dominating solution is found or the whole neighborhood has been explored if there is no dominating solution. The neutral neighbors found in Step 3 become part of the candidate solution set $\mathcal{C}$ to update the archive. In Step 4 these neutral neighbors could be included in the archive only if they are non dominated by all members of the current archive. Then in Step 2 of the next iteration the newly found neutral neighbors can be selected for exploration. Thus, DMLS algorithms in order to exploit a neutral neighbor of x also requires that it is non-dominated by the archive.

Neutrality seems to be exploited to increase the performance of DMLS as equivalent neighbors may be candidates to be archived. However, it is not clear the contribution of neutral neighbors to the effectiveness of DMLS algorithms. In this paper, we want to clarify and show the impact of using neutral neighbors in multi-objective local search. To do so, we will analyze two configurations of DMLS denoted DMLS $(1 \cdot \overline{1_{\succ}})$ and DMLS $(\bigstar \cdot \overline{1_{\succ}})$ where neutrality is never exploited and compare them with already existing strategies for DMLS algorithms that explore to some degree neutrality. In DMLS $(1 \cdot \overline{1_{\succ}})$ and DMLS $(\bigstar \cdot \overline{1_{\succ}})$ the neighborhood of each selected solution is explored until a dominating solution is found and only this neighbor represents a candidate to be archived, thus never exploiting the neutral neighbors of a solution.

3.4 New Neutrality-based Strategies

In addition to configurations of DMLS, proposed by Liefooghe et al., where neutrality could be implicitly exploited if neutral neighbors are non-dominated by the archive, we propose in this paper two new configurations where neutrality can be exploited in two different steps of the algorithm: either during the exploration of the neighborhood or in the formation of the candidate set of solutions to be archived.

In DMLS $(1 \cdot 1_{\not\succ})$ [4], neutrality can be exploited when the first non-dominated neighbor found during exploration of the neighborhood happens to be a neutral neighbor and later it is non-dominated by the archive. In the $1_{\not\succ}$ exploration strategy, the first non-dominated solution found could be either a neutral neighbor or a dominating neighbor. It is arguable whether the first neutral neighbor found would be the best to improve later the Pareto front in the archive, so that neutrality could be exploited. Similarly, it is also arguable whether the first dominating neighbor could be the best dominating neighbor. Therefore, we propose a $k_{\nprec}$ exploration strategy, where the neighborhood of a solution is explored until k different non-dominated neighbors have been found. This strategy gives the opportunity to explore more widely the neighborhood of a solution. Indeed more chance to find one or more dominating neighbors is given. In addition, since all non-dominated neighbors are collected in the candidate set $\mathcal{C}$ to update the archive, this strategy increases the likelihood of finding neutral neighbors that can become part of the archive, diversifies the Pareto front, and emphasizes the exploitation of neutrality. The new DMLS with the $k_{\nprec}$ exploration strategy is denoted DMLS $(1 \cdot k_{\nprec})$, where the number k is an integer defined from 1 to the neighborhood size.

The candidate set of solutions $\mathcal{C}$ considered to update the archive is a key element when dealing with neutrality. Indeed, Liefooghe et al. take into account all neutral neighbors visited during the neighborhood exploration when a $1_{\succ}$ strategy is used. In this paper, in addition to collect neutral neighbors in the candidate set $\mathcal{C}$ of solutions to update the archive, we propose to use some of them for further exploration, before they are used to update the archive.

DMLS $(\bigstar \cdot 1_{\succ})$ [4] is a configuration where all solutions of the archive are selected to be explored until a dominating neighbor is found for each one. This algorithm may integrate a large number of solutions in $\mathcal{C}$ during a single step of archiving. We modify Step 3 of the DMLS $(\bigstar \cdot 1_{\succ})$ algorithm. When a non-dominated neighbor x' is found, we check Pareto dominance between x' and those already in the set $\mathcal{X}$ selected for exploration. If no solution in $\mathcal{X}$ is dominated by x', then x' is added to the exploration set $\mathcal{X}$. Thus, the exploration set $\mathcal{X}$ grows dynamically as neutral neighbors are found. This strategy allows to explore neutral neighbors that could not be archived in Step 4 after finding the dominating neighbors of solutions in $\mathcal{X}$. This proposed DMLS, denoted DMLS $(\bigstar + \mathcal{X}_{\nprec} \cdot 1_{\succ})$, takes advantage of neutrality more intensively than the known configurations of DMLS. Figure 3 gives the complete nomenclature of DMLS algorithms with the proposed configurations (in the red boxes) and the most used DMLS configurations. Note that in this figure the definition of the candidate set of solutions $\mathcal{C}$ used to update the archive is explicitly described.

4 Experiments and Discussion

The aim of this section is to compare performance of the different DMLS configurations studied in this work and clarify the effects of neutrality on the Permutation Flowshop Scheduling Problem (PFSP).

4.1 Experimental Protocol

Instances Experiments are realized on classical Muti-objective PFSP instances. These instances have been proposed by Minella et al. [8] as an extension of the well-known random generated instances of Taillard [9], in which due dates have been added. In the following, these instances are denoted by a triplet $(JJJ \times$

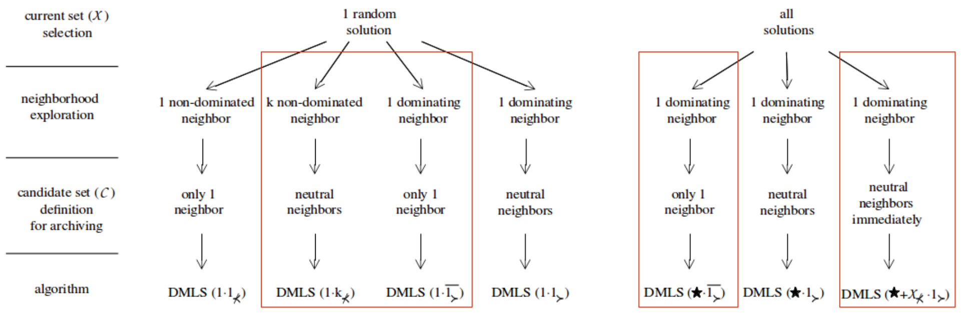
current set ($\mathcal{X}$) selection
neighborhood exploration
candidate set ($\mathcal{C}$) definition for archiving
algorithm
1 random solution
all solutions
1 non-dominated neighbor
k non-dominated neighbor
1 dominating neighbor
1 dominating neighbor
1 dominating neighbor
1 dominating neighbor
1 dominating neighbor
only 1 neighbor
neutral neighbors
only 1 neighbor
neutral neighbors
only 1 neighbor
neutral neighbors
neutral neighbors immediately
DMLS ($1 \cdot 1_{\nprec}$)
DMLS ($1 \cdot k_{\nprec}$)
DMLS ($1 \cdot \overline{1_{\succ}}$)
DMLS ($1 \cdot 1_{\succ}$)
DMLS ($\bigstar \cdot \overline{1_{\succ}}$)
DMLS ($\bigstar \cdot 1_{\succ}$)
DMLS ($\bigstar + \mathcal{X}_{\nprec} \cdot 1_{\succ}$)

Fig. 3. Nomenclature of DMLS algorithms with the proposed configurations.

$MM \times NN$), where JJJ is the number of jobs, MM is the number of machines, and NN is the identifier of the ($JJJ \times MM$) instance.

Performance Assessment In order to rank the different algorithms and observe the behavior and influence of the neutral neighbors on performance, three complementary indicators are used as recommended in [3]. Namely, unary ε-indicator $I^1_{\varepsilon+}$, hypervolume difference indicator I^-_H, and Spread. These indicators are based on set Z^{all} that is the union of the final sets of solutions obtained by all algorithms and on the reference set R that contains the Pareto set of Z^{all}. The three performance indicators are explained below, where A stands for the set of solutions found by an algorithm.

ε-indicator $I^1_{\varepsilon+}$ The unary ε-indicator is computed using the binary version given by (4) and the reference set R, with $I^1_{\varepsilon+}(A) = I_{\varepsilon+}(A, R)$.

$$I_{\varepsilon+}(A, R) = \inf_{\varepsilon \in \mathbb{R}} \{\forall z^1 \in R, \exists z^2 \in A, \forall i \in 1 \dots n, z^1_i \leq \varepsilon + z^2_i\} \tag{4}$$

Hypervolume difference indicator I^-_H The hypervolume indicator I_H is computed by the measure of the hypervolume between a set of solutions and the point $z = (z_1, \dots, z_n)$ where z_k is the upper bound of the k^{th} objective considering all solutions of Z^{all}. The hypervolume difference indicator I^-_H is then computed with $I^-_H(A) = I_H(R) - I_H(A)$.

Spread The spread indicator used in this paper is computed as follows. First, the two solutions of Z^{all} that reach the extrema relatively to the two objectives are selected, and filtered out of the set of the considered solutions. Given d_f and d_l the distances to those extreme points, $\bar{d}$ the mean of the distances, and d_i the distance between solutions of the set, the spread indicator is given by (5).

$$\Delta = \frac{d_f + d_l + \sum |d_i - \bar{d}|}{d_f + d_l + \sum \bar{d}} \tag{5}$$

Experimental design All DMLS implementations are realized under the ParadisEO 2.0 software framework [5]. Most of the performance assessment indices are computed using the PISA platform [2] and its performance assessment module. The spread indicator has been developed to be automatically computed into PISA. The results are then verified with the *Friedman* statistical test, and a global ranking is computed using the *Wilcoxon* statistical test.

Seven instances have been selected from Minella et al., spanning over seven problem sizes. The seven algorithms of Figure 3 are compared. For the parameter k in DMLS $(1 \cdot k_{\nprec})$, two different values *low* and *high* have been tested, leading to two versions of this algorithm: DMLS $(1 \cdot k^{low}_{\nprec})$ and DMLS $(1 \cdot k^{high}_{\nprec})$. The *low* and *high* values of k depend on the number of jobs of the instance as the size of the neighborhood depends on it. Parameter k has been set arbitrarily according to the number of jobs: $k = 5$ and 10 for 20 jobs, $k = 15$ and 25 for 50 jobs, and $k = 20$ and 50 for 100 jobs.

For each instance, 20 executions have been recorded for each algorithm. A maximal runtime has been fixed for each size of instance corresponding to $N \times M$ seconds. Those runtimes were sufficient for all algorithms to converge, even if they did not reach a natural termination.

4.2 Experimental Results

Table 1 shows the rankings computed with the indicator $I^1_{\varepsilon+}$ for each instance with respect to the final Pareto local set R. Similarly, Table 2 and Table 3 show the rankings computed with I^-_H indicator and *spread* indicator, respectively.

The Friedman statistical tests give a p-value of $2.449e^{-6}$ for the $I^1_{\varepsilon+}$ indicator, $4.796e^{-6}$ for I^-_H and $1.758e^{-5}$ for *spread*. Thus, the behavior of the all algorithms is statistically different on the three indicators, and ranks give valuable information about performance. These tables allow several observations.

Table 1. Rankings according to $I^1_{\varepsilon+}$

Instance	$(1 \cdot \overline{1_{\succ}})$	$(\star \cdot \overline{1_{\succ}})$	$(1 \cdot 1_{\nprec})$	$(1 \cdot k^{low}_{\nprec})$	$(1 \cdot k^{high}_{\nprec})$	$(1 \cdot 1_{\succ})$	$(\star \cdot 1_{\succ})$	$(\star + \mathcal{X}_{\nprec} \cdot 1_{\succ})$
$(020 \times 05 \times 01)$	7	8	**1**	5	2	6	4	3
$(020 \times 10 \times 01)$	8	7	2	6	5	4	3	**1**
$(020 \times 20 \times 01)$	8	7	4	5	2	6	3	**1**
$(050 \times 10 \times 01)$	7	8	2	4	5	6	3	**1**
$(050 \times 20 \times 01)$	8	7	5	2	4	6	3	**1**
$(100 \times 10 \times 01)$	7	8	3	5	4	6	2	**1**
$(100 \times 20 \times 01)$	7	8	3	6	5	4	2	**1**
mean	7.42	7.57	2.85	4.71	3.85	5.42	2.85	1.28

No neutrality exploitation Tables 1, 2 and 3 show that algorithms DMLS $(1 \cdot \overline{1_{\succ}})$ and DMLS $(\star \cdot \overline{1_{\succ}})$ share the worse results on the three indicators. These two methods select during the neighborhood exploration one single dominating neighbor for each explored solution, without any neutrality consideration,

Table 2. Rankings according to I_H^-

Instance	$(1\cdot\overline{1_\succ})$	$(\bigstar\cdot\overline{1_\succ})$	$(1\cdot 1_{\not\prec})$	$(1\cdot k^{low}_{\not\prec})$	$(1\cdot k^{high}_{\not\prec})$	$(1\cdot 1_\succ)$	$(\bigstar\cdot 1_\succ)$	$(\bigstar+\mathcal{X}_{\not\prec}\cdot 1_\succ)$
$(020\times 05\times 01)$	8	7	3	4	**1**	6	5	2
$(020\times 10\times 01)$	8	7	2	6	5	4	3	**1**
$(020\times 20\times 01)$	8	7	4	5	2	6	3	**1**
$(050\times 10\times 01)$	7	8	2	4	5	6	3	**1**
$(050\times 20\times 01)$	8	7	5	2	4	6	3	**1**
$(100\times 10\times 01)$	7	8	3	6	4	5	2	**1**
$(100\times 20\times 01)$	7	8	3	6	5	4	2	**1**
mean	7.57	7.42	3.14	4.71	3.71	5.28	3.00	1.14

Table 3. Rankings according to the spread indicator

Instance	$(1\cdot\overline{1_\succ})$	$(\bigstar\cdot\overline{1_\succ})$	$(1\cdot 1_{\not\prec})$	$(1\cdot k^{low}_{\not\prec})$	$(1\cdot k^{high}_{\not\prec})$	$(1\cdot 1_\succ)$	$(\bigstar\cdot 1_\succ)$	$(\bigstar+\mathcal{X}_{\not\prec}\cdot 1_\succ)$
$(020\times 05\times 01)$	8	7	3	5	6	4	**1**	2
$(020\times 10\times 01)$	7	8	3	6	2	4	5	**1**
$(020\times 20\times 01)$	8	7	4	6	3	5	2	**1**
$(050\times 10\times 01)$	7.5	7.5	6	3	5	4	**1**	2
$(050\times 20\times 01)$	8	7	6	5	3	4	2	**1**
$(100\times 10\times 01)$	8	7	3	6	5	4	**1**	2
$(100\times 20\times 01)$	8	7	6	5	4	3	2	**1**
mean	7.78	7.21	4.42	5.14	4.00	4.00	2.00	1.42

i.e. not even one neutral neighbor is considered as candidate solution to update the archive. If this strategy allows to quickly optimize at the beginning of the search, it does not allow to obtain a good approximation of the whole Pareto front.

Considering neutrality to update the archive One way to take profit from neutrality during the search is, as exposed before, to collect neutral neighbors during the neighborhood exploration and use them as candidates to update the archive. A strategy may consist in exploring the neighborhood until a dominating neighbor is reached (as in the worst versions), but keeping all equivalent neighbors encountered. This leads to DMLS $(1\cdot 1_\succ)$ and DMLS $(\bigstar\cdot 1_\succ)$. Note that DMLS $(1\cdot 1_\succ)$ is ranked better than DMLS $(1\cdot\overline{1_\succ})$ and DMLS $(\bigstar\cdot 1_\succ)$ is ranked better than DMLS $(\bigstar\cdot\overline{1_\succ})$. This shows that considering neutrality in the candidate set to update the archive seems to be effective. Methods DMLS $(1\cdot 1_{\not\prec})$, DMLS $(1\cdot k^{low}_{\not\prec})$ and DMLS $(1\cdot k^{high}_{\not\prec})$ also consider neutral neighbors as interesting neighbors to update the archive and explore the neighborhood until one or several (k) non-dominated solutions (i.e. either neutral or dominating solutions) are found. Results show that these strategies perform better than methods that only consider the first dominating neighbor encountered. Moreover, when k neighbors are required, several potential interesting solutions may become part of the candidate set to update the archive, which improves the quality of the obtained approximation of the Pareto optimal set. Note that a high value of k leads to better results.

Considering neutrality before archiving Another way to exploit neutral neighbors is to add them dynamically to the set of solutions $\mathcal{X}$ to be explored in addition to the set of candidate solutions to update the archive, as explained in section 3.3. This leads to the method DMLS ($\bigstar + \mathcal{X}_{\not\prec} \cdot 1_{\succ}$) which obtained the best results over all the tested methods. This method outperforms the method DMLS ($\bigstar \cdot 1_{\succ}$) that also takes into account neutral neighbors to update the archive. This is because the dynamical insertion of neutral neighbors into the set of solutions to be explored allows the method to go deeper in the search. In addition, it also saves some computational effort.

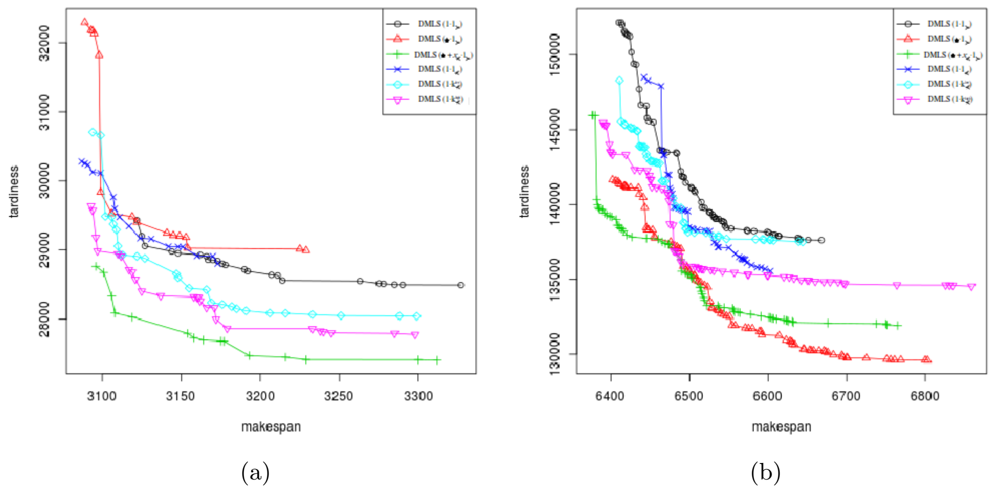

Fig. 4. Pareto Fronts for the instances $050 \times 10 \times 01$ (a) and $100 \times 20 \times 01$ (b).

These above observations are complemented by Figure 4 (a) and (b) that show the Pareto fronts obtained by each method on two instances. These figures illustrate the good average performance of DMLS ($\bigstar + \mathcal{X}_{\not\prec} \cdot 1_{\succ}$) and show that DMLS ($\bigstar \cdot 1_{\succ}$) is able to produce very good results on large instances. These two figures allow to confirm also that DMLS ($1 \cdot k_{\not\prec}^{high}$) outperforms DMLS ($1 \cdot k_{\not\prec}^{low}$) as indicated on the previous tables.

In summary, these experiments show that non considering neutral neighbors (method DMLS ($1 \cdot \overline{1_{\succ}}$) and DMLS ($\bigstar \cdot \overline{1_{\succ}}$)) is less efficient than considering them. In particular, the diversity of the Pareto front produced is greatly impacted. Also, as shown by the not so good performance of method DMLS $(1\cdot 1_{\not\prec})$, in particular in terms of spread, the first found neutral neighbors are not always of good quality and it may be important to consider several of them in order to improve results. Moreover, the exploitation of neutral neighbors that may be dominated by the archive could lead to improve the performance of local search Pareto dominance based approaches.

5 Conclusion

Neutrality has obviously a critical role in multi-objective combinatorial optimization, and furthermore in local search algorithms. Small changes in the way neutral neighbors are handled greatly modify the general behavior of algorithms. This is why the understanding of the relation between local search and neutrality is very important in multi-objective as well as in single-objective optimization. This paper extended the concept of neutrality to multi-objective optimization, focused the discussions about the neutrality in the context of dominance-based multi-objective local search algorithms, and proposed new strategies to improve the behavior of those algorithms towards the exploitation of neutral neighbors. We verified the proposed strategies on a classical bi-objective problem. Experiments showed overall the advantage of exploiting neutral neighbors. It also showed the importance of considering a set of neutral neighbors, instead of a single one, in order to increase the performance in term of diversity and convergence.

However, as it was shown by the not so good performance of method DMLS $(1\cdot 1_{\not\prec})$, first found neutral neighbors may not be of good quality, and it could be interesting, not only to consider several of them, but to select some of them. This is one of the further question we want to address. Another interesting question, is to analyze how this neutrality concept may be transposed to multi-objective problems with more than two objectives, as the number of neutral neighbors may increase significantly with the number of objectives. Additionally, it will be interesting to study neutrality under other classes of fitness assignment methods in multi-objective optimization.

References

1. Barnett, L.: Netcrawling - optimal evolutionary search with neutral networks. In: Proceedings of the 2001 Congress on Evolutionary Computation. pp. 30–37. CEC 2001, IEEE Press (2001)
2. Bleuler, S., Laumanns, M., Thiele, L., Zitzler, E.: Pisa: A platform and programming language independent interface for search algorithms. In: EMO. pp. 494–508 (2003)
3. Knowles, J., Thiele, L., Zitzler, E.: A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich (Feb 2006)

4. Liefooghe, A., Humeau, J., Mesmoudi, S., Jourdan, L., Talbi, E.G.: On dominance-based multiobjective local search: design, implementation and experimental analysis on scheduling and traveling salesman problems. J. Heuristics 18(2), 317–352 (2012)
5. Liefooghe, A., Jourdan, L., Talbi, E.G.: A software framework based on a conceptual unified model for evolutionary multiobjective optimization: Paradiseo-moeo. European Journal of Operational Research 209(2), 104–112 (2011)
6. Lourenco, H., Martin, O., Stutzle, T.: Iterated local search. In: Glover, F., Kochenberger, G. (eds.) Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol. 57, pp. 321–353. Kluwer Academic Publishers, Norwell, MA (2002)
7. Marmion, M.E., Dhaenens, C., Jourdan, L., Liefooghe, A., Vérel, S.: Nils: A neutrality-based iterated local search and its application to flowshop scheduling. In: EvoCOP. pp. 191–202 (2011)
8. 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)
9. Taillard, E.: Benchmarks for basic scheduling problems. European Journal of Operational Research 64(2), 278–285 (1993)
10. Verel, S., Collard, P., Clergue, M.: Scuba Search : when selection meets innovation. In: Evolutionary Computation, 2004. CEC2004 Evolutionary Computation, 2004. CEC2004. pp. 924 – 931. IEEE Press, Portland (Oregon) United States (2004)