Handling Priority Levels in Mixed Pareto-Lexicographic Many-Objective Optimization Problems*

Leonardo Lai[1][0000−0003−2219−0557], Lorenzo Fiaschi[1][0000−0002−5413−0148], Marco Cococcioni[1][0000−0002−7020−1524], and Kalyanmoy Deb[2][0000−0001−7402−9939]

[1] Pisa University, Largo Lucio Lazzarino 1 – 56122, Italy
leonardo.lai@live.com, lorenzo.fiaschi@phd.unipi.it, marco.cococcioni@unipi.it
[2] Michigan State University, East Lansing, MI 48824, USA
kdeb@egr.msu.edu
COIN Report Number 2020025

Abstract. This paper studies a class of mixed Pareto-Lexicographic multi-objective optimization problems where the preference among the objectives is available in different *priority levels* (PLs) before the start of the optimization process – akin to many practical problems involving domain experts. Each priority level (PL) is a group of objectives having an identical importance in terms of optimization, so that they must be optimized in the standard Pareto sense. However, between two PLs, a lexicographic preference structure exists. Clearly, finding the entire set of Pareto optimal solutions first and then choosing the lexicographic solutions using the given PL structure is not computationally efficient. A new efficient algorithm is presented here using a recent mathematical breakthrough in handling infinite and infinitesimal quantities: the *Grossone* methodology. The proposal has been implemented within a popular multi-objective optimization algorithm (NSGA-II), thereby obtaining its generalized version named PL-NSGA-II, although other EMO or EMaO algorithms could have also been used instead. A quantitative comparison of PL-NSGA-II performance against existing algorithms is made. Results clearly show the advantage of the proposed Grossone-based methodology in solving such priority-level many-objective problems.

Keywords: Multi-Objective Optimization · Lexicographic Optimization · Evolutionary Computation · Grossone Methodology

1 Introduction

Multi- and many-objective optimization represents a well-studied and active research topic to solve problems that appear in several areas of engineering and

* Work partially supported by the Italian MIUR – CrossLab project (Departments of Excellence) and partially by the University of Pisa funded project PRA_2018_81 "Wearable sensor systems: personalized analysis and data security in healthcare".

economics. As the problem dimensionality grows, computational issues arise: ineffectiveness of the standard Pareto dominance and conventional recombination operators, necessity to maintain a significantly larger population, non-linear increase of the required computational power, difficulty to visualize the solutions. Optimization problems that originate from real scenarios often manifest some kind of priority relation among their objectives, since they may not be all of equal preference or interest in practice. In these cases, the objectives can be possibly reorganized based on their preferences, still including all of them in the problem formulation.

The present work introduces a novel way to deal with multi- and many-objective real-world problems exploiting the priority information in a structured way. Specifically, it focuses on *Mixed Pareto-Lexicographic Multi-objective Optimization Problems* (abbreviated as MPL-MOPs), a broad family of problems that assume the existence of priority structure among objectives. A first subclass, characterized by objectives aggregated by priority as "chains", has already been investigated by the authors [14]. This article, instead, considers a category of problems whose objectives can be arranged in priority levels (PL-MPL-MOPs). The common element between this and the previous study is the use of the so-called *Grossone Methodology* (GM) [20], a mathematical framework enabling one to deal with infinite and infinitesimal quantities, in this case the weights by means of which levels are assigned priority.

Section 2 outlines the mathematical description of the problems of our interest, Section 3 covers the essential elements of the GM, providing the basic knowledge to deal with PL-MPL-MOPs. Section 4 describes how GM helps solving them, dealing with the mathematical reformulation of the model and the definition of a new and more general dominance relation. Section 5 introduces a Grossone-based generalization of NSGA-II [10], called by the authors PL-NSGA-II, which makes the algorithm able to cope with PL-MPL-MOPs by leveraging the original ideas of this work. Finally, Section 6 presents a quantitative comparison of PL-NSGA-II and four other EMO algorithms on PL-variants of two popular benchmarks and an engineering problem. The results confirm the benefits achieved by including the priority information in the optimization procedure. Conclusions are finally drawn in Section 7.

2 Mixed Pareto-Lexicographic Problems (MPL-MOPs)

Purely Pareto or purely lexicographic multi-objective optimization problems (MOPs) are commonly found in the literature. The former are analytically formulated like in Eqn. (1), where m is the number of objective functions, Ω is the feasible decision space (or search space), and the optimization is performed in the standard Pareto-sense. On the other hand, lexicographic optimization complies with the model in Eqn. (2), where the optimization involves scalarizing by means of a lexicographic ranking of the functions, that is, f_1 is infinitely more important than f_2 ($f_1 \gg f_2$), which in turn has priority over f_3 ($f_2 \gg f_3$), and so on.

$$\begin{aligned} &\min \{f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}),\} \\ &\text{s.t. } \mathbf{x} \in \Omega. \end{aligned} \quad (1) \qquad \left| \qquad \begin{aligned} &\text{lexmin } \{f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})\}, \\ &\text{s.t. } \mathbf{x} \in \Omega. \end{aligned} \right. \quad (2)$$

Interestingly, both models represent edge cases of a more general family of problems, which has received little attention in the literature: Mixed-Pareto-lexicographic MOPs (MPL-MOPs). The mathematical description is:

$$\begin{aligned} &\min \{f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})\}, \\ &\text{s.t. } \mathbf{x} \in \Omega, \\ &\quad \wp(f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})) \text{ supplied}, \end{aligned} \quad (3)$$

where $\wp(\cdot)$ is a generic distribution of priorities among the objectives. The problem of dealing with arbitrary priority policies is crucial for further improvements of EMO algorithms, since routines able to exploit priority information would definitely help in finding better solutions, as suggested in [12]. In the literature, some studies concerning ad-hoc proposal to cope with priority relations exist, such as ε-constrained methods [11] or scalarization [16]. In spite of their superior performance, they are not specifically designed to deal with MPL-MOPs, resulting in relevant drawbacks such as the inability to find multiple solutions per run, or the need for an appropriate choice of scalarization weights beforehand.

The class of MPL-MOPs is too heterogeneous to design a one-fits-all approach; instead, it is much more reasonable to narrow the focus on specific instances where the shape of the $\wp(\cdot)$ function has peculiar properties or is particularly significant to model real-world scenarios. In this work, $\wp(\cdot)$ groups the objective in priority levels (PLs) and the ordering relation is defined on the levels rather than the functions, as illustrated in Figure 1. The key idea is: i) each group clusters objectives by importance; ii) a group contains only objectives having the same priority.

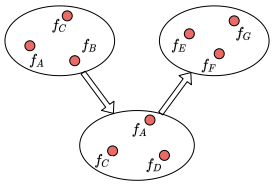

Fig. 1: PL-MPL-MOP structure.

The mathematical formulation of a generic description of a PL-MPL-MOP is:

$$\text{lexmin} \left[\min \begin{pmatrix} f_1^{(1)}(\mathbf{x}) \\ \vdots \\ f_{m_1}^{(1)}(\mathbf{x}) \end{pmatrix}, \min \begin{pmatrix} f_1^{(2)}(\mathbf{x}) \\ \vdots \\ f_{m_2}^{(2)}(\mathbf{x}) \end{pmatrix}, \ldots, \min \begin{pmatrix} f_1^{(p)}(\mathbf{x}) \\ \vdots \\ f_{m_p}^{(p)}(\mathbf{x}) \end{pmatrix} \right], \quad (4)$$

where p is the number of PLs and $f_i^{(j)}$ is the i-th objective in the j-th PL. Note that an objective can repeat in multiple PL. In such problems, the Pareto-optimal solutions of the objectives in the first PL form the decision space for the Pareto-optimization of those in the second PL, and so on. Notice that a purely Pareto optimization problem is a special case of PL-MPL-MOP, where only one PL is considered, while a lexicographic problem is one with multiple PLs having at least one objective in each. In the next section, a possible numerical way to deal with this kind of problems on a computer is presented, as there do not exist tools to concurrently perform Pareto and lexicographic optimization yet.

3 The Grossone Methodology

Grossone Methodology (GM) [20] is a numerical framework that makes it possible to work with infinite and infinitesimal numbers. It has already been successfully applied to many different optimization problems [3–8, 14]. The GM fundamental element is the infinite unit *Grossone*, denoted by ①, which allows one to build numerical values composed by finite, infinite and infinitesimal components, known as *gross-scalar* (G-scalar in brief). The latter are indicated as:

$$c = c_{p_m}①^{p_m} + \ldots + c_{p_0}①^{p_0} + \ldots + c_{p_{-k}}①^{p_{-k}},$$

where $m, k \in \mathbb{N}$, the exponents p_i and the digits $c_{p_i} \neq 0$ are called *gross*-powers (G-powers) and *gross*-digits (G-digits), respectively.

The four basic operations between two G-scalars are reasonably intuitive, easy to implement and inherit all the standard properties like associativity, commutativity and existence of the inverse. For instance, consider the following few lines to get a basic understanding of their arithmetical behavior, which is similar to the one of polynomials:

$$①\cdot(①+2) = ①^2 + 2①, \qquad 0 < \frac{1}{①} = ①^{-1} < ①^0 = 1 < ①^1 = ①,$$
$$\frac{-10.0①^3 + 16.0 + 42.0①^{-3}}{5.0①^3 + 7.0} = -2.0 + 6.0①^{-3}.$$

In addition, the operator $<$ induces a total ordering among the set of G-scalars, and the comparison is made considering the G-powers in descending order: if they differ or they are equal but the corresponding G-digits are different, then the biggest corresponds to the largest G-scalar; otherwise, the next pair of G-powers and G-digits is checked, and so on.

For the rest of the paper, we only consider G-scalars having a finite number of components, each with real-valued G-power and G-digit. In such a scenario, finite numbers are represented by G-scalars with the highest G-power equal to zero, infinitesimal ones with negative highest G-power, and infinite ones with a positive highest G-power. For instance, $-3.4 = -3.4①^0$ and $2 + 0.5①^{-2}$ are both finite numbers, $\pi①^{-1} - ①^{-\sqrt{2}}$ is infinitesimal, while $3①^{\pi} - 70①^{-e}$ is infinite.

4 Use of Grossone in PL-MPL-MOPs

This section shows how ① helps in the design of higher priority-aware routines to deal with PLs better. First, PL-MPL-MOPs are reformulated according to Grossone-based representation where the value of the overall objective function, comprehensive of the Paretian and lexicographic components altogether, can be numerically evaluated. The second and richer part, instead, investigates a novel definition of dominance that possibly fits better the nature of PL-MPL-MOPs. Needless to say, the latter leverages ① as well.

4.1 Problem Reformulation

Similarly to the case of MPL-MOPs with priority chains [14], the lexicographic relation between PLs can be represented both mathematically and in a computer by means of ①. Here, on the other hand, a lexicographic priority between two levels indicates that one group of objectives is infinitely more important than another one. The key idea is to use decreasing powers of ① to represent such ordering, and split up the objectives in well-defined PLs. In mathematical terms, the problem in (4) is reformulated with Grossone as:

$$\min \left[\begin{pmatrix} f_1^{(1)}(\mathbf{x}) \\ \vdots \\ f_{m1}^{(1)}(\mathbf{x}) \end{pmatrix} + ①^{-1} \begin{pmatrix} f_1^{(2)}(\mathbf{x}) \\ \vdots \\ f_{m2}^{(2)}(\mathbf{x}) \end{pmatrix} + \ldots + ①^{1-p} \begin{pmatrix} f_1^{(p)}(\mathbf{x}) \\ \vdots \\ f_{mp}^{(p)}(\mathbf{x}) \end{pmatrix} \right]. \tag{5}$$

In practice, the lexicographic optimization of the priority levels has been transformed into a minimization problem over the weighted average of the PLs contributions. The weights are assigned to the levels by importance: the one with the highest priority is weighted by $①^0$ (the digit, 1, is omitted for brevity), the second one by $①^{-1}$, and so on, until no more PLs remain.

4.2 A Novel Definition of Dominance for PL-MPL-MOPs

When dealing with a PL-MPL-MOP, the Pareto dominance is not suitable because it is oblivious of the priority information. Defining an appropriate *PL-dominance* is not trivial, as demonstrated by the following example, that would be a straightforward but actually a broken definition of PL-Dominance:

Definition 1 (PL-Dominance (INCORRECT)). *Let x and y be two solutions of a PL-MPL-MOP with n PLs: $(\wp_1, \ldots, \wp_n)$ arranged in the order of reducing priority. Let $\wp_i^z$ indicate the i-th PL of solution z, and $x \prec y$ indicates that x Pareto-dominates y. Then, x is said to "PL-dominate" y ($x \prec_{PL} y$) $\iff$ $\exists\, i$ such that $\wp_i^x \prec \wp_i^y$ and $\nexists\, j < i$ such that $\wp_j^y \prec \wp_j^x$.*

This definition surprisingly lacks the transitivity property, i.e. there may exist three elements x, y, z such that x PL-dominates y and y PL-dominates z, but z PL-dominates x. The following counterexample is provided for clarity. Consider the following three solutions x, y, z of a PL-MPL-MOP with two PLs, each with two objectives to be minimized:

$$x = \begin{pmatrix} 1 \\ 6 \end{pmatrix} + ①^{-1} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \qquad y = \begin{pmatrix} 2 \\ 1 \end{pmatrix} + ①^{-1} \begin{pmatrix} 2 \\ 4 \end{pmatrix}, \qquad z = \begin{pmatrix} 0 \\ 5 \end{pmatrix} + ①^{-1} \begin{pmatrix} 5 \\ 7 \end{pmatrix}$$

With reference to Definition 1, there is a dominance-loop since: i) $x \prec_{\mathrm{PL}} y$ because, although they are non-dominated for the first PL, x performs better than y on the second one; ii) $y \prec_{\mathrm{PL}} z$ for the same reason; iii) $z \prec_{\mathrm{PL}} x$ because it does better on the first PL.

The definition of dominance proposed in this paper leverages the concept of *non-dominated sub-fronts*, a generalization of the notion of fronts which consists in nested fronts generated by the priority. The procedure to partition the

population into sub-fronts starts considering only the objectives with the highest priority, and generating a first set of non-dominated fronts as usual, according to Pareto dominance. Then, each front is considered individually, and further split on the basis of the objectives within the second PL, determining new nested fronts (namely, sub-fronts). The same procedure is repeated recursively on the newly-obtained sub-fronts, until no more PLs remain; the result is a population partitioned in a tree-like hierarchy of sub-fronts. Exploiting the natural ordering of the sub-fronts, one can adopt the GM to uniquely identify each leaf-front, i.e. assign an index to each sub-front containing solutions. Such index, which is a G-scalar, is computed as the sum of n components of the form $d_i①^{p_i}$, where n is the number of PLs, p_i is the G-power associated to the i-th PL (see Equation (5)), and d_i is the position of the parent front at the $1-i$ level of the hierarchy. For instance, F_i where $i = 3 + 2①^{-1} + 5①^{-2}$ denotes all the solutions in the front 3 for the primary objectives, front 2 for the secondary objectives (within the individuals in the front 3 for the primaries) and front 5 for the tertiaries (among those with primary front 3 and secondary front 2).
Given such calculation of sub-fronts, the new definition of PL-dominance is:

Definition 2 (PL-Dominance (CORRECT)). *Let x and y be two solutions such that $x \in F_i$ and $y \in F_j$. Then, $x \prec_{PL} y \iff i < j$.*

Notice that $\prec_{\text{PL}}$ is not just a function of two arguments (solutions), but has a global dependency on all the other individuals, since it previously requires the assignment of the fitness rank to the whole population.

Going back to the previous counterexample, assuming for simplicity that there are no more elements in the population, it turns out that x, y and z belong to sub-fronts indexed by $2 + 1①^{-1}$, $1 + 1①^{-1}$ and $1 + 2①^{-1}$, respectively. Therefore, it holds true that $y \prec_{\text{PL}} z$ and $z \prec_{\text{PL}} x$, but it becomes false that x dominates y ($x \not\prec_{\text{PL}} y$). The proof of transitivity of the PL-dominance comes straightforwardly from the definition, and it is omitted here for brevity.

5 Proposed PL-NSGA-II Procedure

In this section, the well-known NSGA-II [10] algorithm is enhanced with PL-dominance and GM in order to implement a generalized version able to effectively reckon with PL-MPL-MOPs too. In practice, one needs to modify NSGA-II four core operations, which are: i) solutions non-dominance rank assignment; ii) population sorting by rank; iii) crowding distance assignment to each individual; iv) selection of the new population from the fronts in best-to-worst order, using crowding distance to break ties. The improved version is called PL-NSGA-II. It is only right to say that the choice to improve NSGA-II rather than MOEA/D or NSGA-III is totally arbitrary, and driven by the simplicity to do it.

The first step improvement is implemented by the routine described in Section 4.2, which assigns a sub-front to each solution. The sub-front index plays exactly the role of a non-dominance rank for the solution. The second step comes directly by the use of GM, which induces a total ordering among G-scalars. Since the non-dominance rank (i.e., the sub-front index) is a G-scalar, one can resort

to the order relation mentioned in Section 3 to do so. Both steps are expected to generate a larger number of smaller-sized fronts, a benign effect since one of the main weaknesses of many-objective algorithms is indeed represented by large sets of non-dominated solutions.

The novel way to compute the crowding distance proposed here follows an approach somewhat similar to the one leading to Equation (5), and it is designed to prioritize solutions that are distant in more important PLs rather than in the lesser ones. Actually, it states that the crowding distance of a solution belonging to a front F_i is the weighted sum of the crowding distances computed at every PL, where the weights are exactly the ones assigned to each of the them. Of course, only the solutions in F_i must be considered for this procedure. Its pseudocode is reported in Algorithm 1.

Since the crowding distance is now a G-scalar too, a total ordering exists and the fourth step is straightforward. The next population is filled with the solutions from the best sub-front, then with those from the second best sub-front, and so on until enough elements have been picked. If a subfront is too large to be included in its entirety, then the solutions with the best crowding distance value are preferred to build the next population.

Altogether, these four improved pieces give birth to PL-NSGA-II, an algorithm which is specifically designed to deal with PL-MPL-MOPs, as well as with standard Pareto and lexicographic problems, being them nothing but corner cases. The next section is devoted to quantitatively assess the performances achieved by the new algorithm PL-NSGA-II.

Algorithm 1 Priority Levels crowding distance assignment.

1: /* *F is the current front in analysis, n is the number of PLs* */
2: **procedure** CROWDING_DIST_ASSIGNMENT(F, n)
3: $p = |F|$
4: **for all** $i \in F$ **do**
5: $CD_i = 0$
6: **for** $q = 1 \dots n$ **do** ▷ For each PL
7: **for** $j = 1 \dots m_q$ **do** ▷ m_q means #objectives in q-th PL
8: $F = sort\left(F, f_j^{(q)}\right)$ ▷ Sort the solutions by the j-th objective
9: $CD_{F[1]}$ += +Inf $①^{1-q}$ ▷ +Inf means "full-scale" (IEEE 754 std.)
10: $CD_{F[p]}$ += +Inf $①^{1-q}$
11: **for** $i = 2 \dots (p-1)$ **do**
12: $CD_{F[i]} \mathrel{+}= ①^{1-q} \frac{f_j^{(q)}(F[i+1]) - f_j^{(q)}(F[i-1])}{f_j^{(q)max} - f_j^{(q)min}}$ ▷ Use of ①
13: **return** CD

6 Results with PL-MPL-MOP

Since there do not exist benchmark problems featuring priority levels yet, the experiments of this section are performed on modified versions of existing problems. We consider two test problems and one engineering design problem here.

The performance of PL-NSGA-II is compared with four EMO algorithms: i) NSGA-II, ii) NSGA-III [19], iii) MOEA/D [21], and iv) GRAPH [17] which can handle supplied preference policies. In order to make the comparison fair, solutions from the non-prioritized algorithms (NSGA-II, NSGA-III, and MOEA/D) are filtered after the optimization on the basis of the supplied PL-dominance information. All algorithms have used the SBX crossover operator and polynomial mutation with a population of 200 individuals and run for 600 generations. For each algorithm, the experiment is repeated 50 times to ensure a statistically stable performance comparison through the metric $\Delta(\cdot) = \max\{IGD(\cdot), GD(\cdot)\}$, where IGD is the *inverted generational distance* and GD indicates the *generational distance* [18] (their computation comes after the normalization of the objective space to the unitary hyper-cube). The use of GD and IGD metrics rather than the hypervolume is due to the fact that a clear way to fairly compute the latter in the case of PL-MPL-MOPs is still missing and under investigation. Concerning MOEA/D, NSGA-II and NSGA-III the implementations from `pymoo` library [1] have been used in this work.

6.1 PL-MaF7 Problem

MaF7 [2] is a widely recognized benchmark problem for multi- and many-objective evolutionary algorithm research. We introduce two priority levels with three objective functions in each level, as follows:

$$\begin{aligned} \min \quad & \left[\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ h(\mathbf{x}) \end{pmatrix} + ①^{-1} \begin{pmatrix} f_1^{(2)}(\mathbf{x}) \\ f_2^{(2)}(\mathbf{x}) \\ h(\mathbf{x}) \end{pmatrix} \right], \\ \text{s.t. } & \mathbf{x} \in [0,\, 1]^{23}, \end{aligned}$$

where $f_1^{(2)}(\mathbf{x}) = -e^{-0.5\, p(\mathbf{x},\, 0.6)}$, $f_2^{(2)}(\mathbf{x}) = |p(\mathbf{x},\, 0.8) - 0.04|$, $p(\mathbf{x},\, c) = (x_1 - c)^2 + (x_2 - c)^2$, $h(\mathbf{x}) = 6 + \frac{27}{20} \sum_{i=3}^{23} x_i - \sum_{i=1}^{2} x_i\, (1 + \sin(3\pi x_i))$.

The first PL has a Pareto front made by four disjoint regions due to periodic function in $h(\mathbf{x})$, as illustrated in Figure 2. The second PL has the effect of isolating just one of them, the one closest to the line $x_1 = x_2 = 1$.

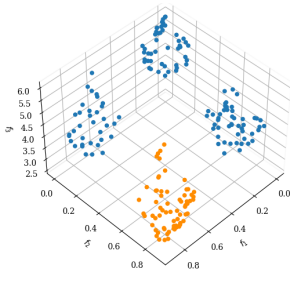

Fig. 2: NSGA-II and true PL-MPL (orange) points for MaF7.

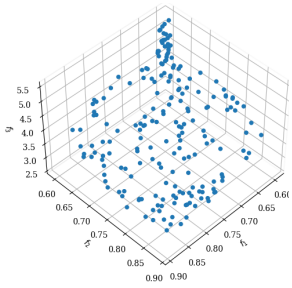

Fig. 3: Obtained PL-NSGA-II points for PL-MaF7.

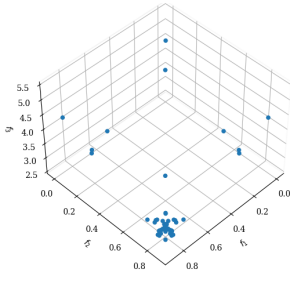

Fig. 4: Obtained MOEA/D (2nd-best) points for PL-MaF7.

The function $f_1^{(2)}$ is a bivariate normal distribution centered in (0.6, 0.6), which assigns higher importance to solutions with (x_1, x_2) closer to the mean point; $f_2^{(2)}$ favors the candidate solutions whose pair (x_1, x_2) is closer to the circumference with center in (0.8, 0.8) and radius 0.2. Finally, the last function selects a single disjoint region: $f_1^{(2)}$ helps keep the optimal part of the front near the origin on the x_1-x_2 plane, whereas h pushes in the opposite direction, i.e., it privileges those points further from the origin.

For each algorithm, the mean and the standard deviation of $\Delta(\cdot)$ are reported in Table 1, ranked from the best to the worst; the best values are boldfaced. Figure 3 reports the primary Pareto front obtained by a single run of PL-NSGA-II, demonstrating the ability of the priority-aware algorithm to accurately generate the right part of the Pareto front, whereas the second-best method (non-priority MOEA/D) cannot find a well-diverse set.

Table 1: Performance ($\Delta(\cdot) = \max\{IGD(\cdot), GD(\cdot)\}$) comparison on MaF7.

Algorithm	**Mean**	**Std. Deviation**	**Avg. # Pareto Sols.**
PL-NSGA-II	$\mathbf{1.025e^{-4}} + 3.451e^{-5}①^{-1}$	$\mathbf{4.201e^{-5}} + 1.788e^{-5}①^{-1}$	**200.00**
GRAPH	$0.332 + 0.558①^{-1}$	$0.038 + 0.079①^{-1}$	**200.00**
MOEA/D	$0.003 + 0.001①^{-1}$	$0.001 + 0.001①^{-1}$	60.00
NSGA-III	$0.357 + 0.543①^{-1}$	$0.021 + 0.034①^{-1}$	18.32
NSGA-II	$0.010 + 0.005①^{-1}$	$0.004 + 0.002①^{-1}$	9.44

6.2 PL-MaF11 Problem

MaF11 [2] is a benchmark problem from the Walking Fish Group (WFG) test suite, named there WFG2. We consider a three-objective version of the problem. The original objective functions, now come together in the first PL, are

$$
\begin{aligned}
f_1^{(1)}(\mathbf{x}) &= y_3 + 2\left(1 - \cos\left(y_1\frac{\pi}{2}\right)\right)\left(1 - \cos\left(y_2\frac{\pi}{2}\right)\right),\\
f_2^{(1)}(\mathbf{x}) &= y_3 + 4\left(1 - \cos\left(y_1\frac{\pi}{2}\right)\right)\left(1 - \sin\left(y_2\frac{\pi}{2}\right)\right),\\
f_3^{(1)}(\mathbf{x}) &= y_3 + 6\left(1 - y_1\cos^2\left(5y_1\pi\right)\right),
\end{aligned}
$$

where

$$
y_i = \begin{cases} (t_i^{(3)} - 0.5)\max(1, t_3^{(3)}) + 0.5, & i = 1, 2,\\ t_3^{(3)}, & i = 3, \end{cases}
\quad
t_i^{(3)} = \begin{cases} t_i^{(2)}, & i = 1, 2,\\ \frac{1}{5}\sum_{j=3}^{7} t_j^{(2)}, & i = 3, \end{cases}
$$

$$
t_i^{(2)} = \begin{cases} t_i^{(1)}, & i = 1, 2,\\ t_{2i-3}^{(1)} + t_{2i-2}^{(1)} + 2|t_{2i-3}^{(1)} - t_{2i-2}^{(1)}|, & i = 3:7, \end{cases}
\quad
t_1^{(1)} = \begin{cases} z_i, & i = 1, 2,\\ \frac{|z_i - 0.35|}{|\lfloor 0.35 - z_i \rfloor| + 0.35}, & i = 3:12, \end{cases}
$$

with $z_i = x_i/2$ and $x_i \in [0, 2i]$ for $i = 1:12$. The second PL, instead, is constructed by the following two objective functions:

$$
f_i^{(2)} = -\left(f_1^{(1)} - 2f_2^{(1)} + (-1)^i\frac{11}{4}\right)^2, \quad i = 1, 2.
$$

It consists in two parabolic cylinders which select only three pieces of the original Pareto front: the middle region (regardless the height) and the two farthest points from the origin on the $f_1^{(1)} - f_2^{(1)}$ plane. It was considered to be a challenging problem, so we use 2,500 generations on a population of 200 individuals to show a satisfying level of convergence [13]. Figure 5 shows the final non-dominated solutions of the original problem found by NSGA-III. Figure 6 and 7 report the

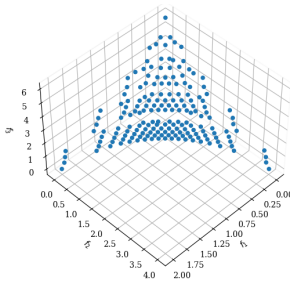

Fig. 5: NSGA-III front of MaF11.

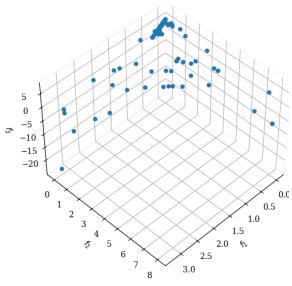

Fig. 6: GRAPH front of MaF11.

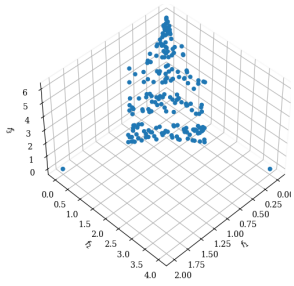

Fig. 7: PL-NSGA-II solutions on PL-MaF11.

optimal front considering both the PLs when approximated by GRAPH and PL-NSGA-II, respectively. Notice that PL-NSGA-II algorithm, being aware of the priority, requires only 600 iterations to achieve such result, in contrast with the 2,500 iterations needed by NSGA-II. Table 2 presents the detailed performance comparison, indicating superior performance of PL-NSGA-II.

Table 2: Performance ($\Delta(\cdot) = \max\{IGD(\cdot), GD(\cdot)\}$) comparison for MaF11.

Algorithm	**Mean**	**Std. Deviation**	**Avg. # Pareto Sols.**
PL-NSGA-II	$\mathbf{0.642} + 0.240①^{-1}$	$\mathbf{0.373} + 0.091①^{-1}$	**200.00**
GRAPH	$0.723 + 0.248①^{-1}$	$0.404 + 0.098①^{-1}$	**200.00**
MOEA/D	$0.661 + 0.360①^{-1}$	$0.370 + 0.074①^{-1}$	3.08
NSGA-II	$0.828 + 0.348①^{-1}$	$0.428 + 0.121①^{-1}$	3.08
NSGA-III	$0.852 + 0.363①^{-1}$	$0.441 + 0.123①^{-1}$	1.52

6.3 Crashworthiness Problem

Finally, we consider the crashworthiness problem [15], which was solved using NSGA-III [9]. Two PLs are formed: the first PL involves minimizing the mass (f_1) and acceleration (f_2), while the second PL minimizes acceleration (f_2) and the toe-board intrusion compliance due to a crash (f_3). This choice reflects the hierarchical interests of a company which seeks to design high performing cars caring also about the driver's safety. Figure 8 shows the obtained PL-NSGA-II

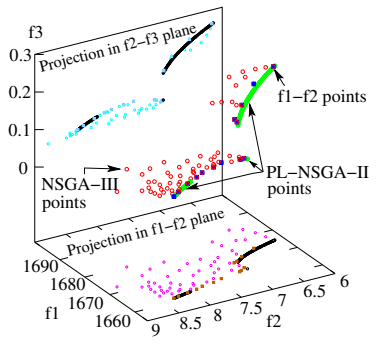

Fig. 8: PL-NSGA-II points lie on edge of f_1-f_2 trade-off points.

points. It clearly shows how PL-NSGA-II points are non-dominated for f_1-f_2 and then non-dominated for f_2-f_3. Table 3 reports the performance of the algorithms, in accordance to the $\Delta(\cdot)$ metric. PL-NSGA-II significantly outperforms the other algorithms, showing a very small standard deviation. In addition, PL-NSGA-II is able to find a higher number of Pareto-optimal solutions than non-prioritized algorithms. This latter aspect is important, since a lot of computational effort is consumed in optimizing towards useless directions. Overall, results shows that including the information about the priority structure directly into the algorithms greatly helps finding more and better solutions, without affecting their generality at all.

Table 3: Performance ($\Delta(\cdot) = \max\{IGD(\cdot), GD(\cdot)\}$) on Crashworthiness.

Algorithm	**Mean**	**Std. Deviation**	**Avg. # Pareto Sols.**
PL-NSGA-II	$\mathbf{2.766e^{-7}} + 3.556e^{-7}①^{-1}$	$\mathbf{9.865e^{-8}} + 1.235e^{-7}①^{-1}$	**200.00**
GRAPH	$0.307 + 0.286①^{-1}$	$0.13 + 0.13①^{-1}$	**200.00**
MOEA/D	$0.158 + 0.293①^{-1}$	$0.022 + 0.001①^{-1}$	39.00
NSGA-II	$0.001 + 0.003①^{-1}$	$0.001 + 0.001①^{-1}$	16.82
NSGA-III	$0.004 + 0.004①^{-1}$	$0.002 + 0.003①^{-1}$	7.48

7 Conclusions

The goal of this paper has been to introduce PL-MPL-MOPs, a sub-category of MPL-MOPs where the objectives are grouped in known levels of priority. Furthermore, the first attempt to improve an already existing algorithm by means of a tool to deal with level-like structured priorities has been presented, implemented and analyzed. It exploits a novel mathematical framework called Grossone methodology – a numerical approach to handle infinite and infinitesimal quantities oriented to scientific calculations. The Grossone and a custom non-dominance definition – the PL-dominance – have enabled a significantly improved performance. This has been demonstrated by illustrative experiments carried out on variations of two standard benchmark problems and an engineering design problem, where a second PL is to turn the problem into a PL-MPL-MOP. The numerical results have demonstrated how the PL-extension of NSGA-II is consistently able to outperform other EMO algorithms of various kinds, some aware of the priority and some not, some designed for many-objective problems and some for multiple objectives only. As a future work, the authors are considering the study of PL-based crossover and mutation operators, as well as the design of more challenging benchmarks for PL-MPL-MOPs. Finally, a deeper analysis of the effectiveness of this technique on real problems is also ongoing.

References

1. Blank, J., Deb, K.: Pymoo: Multi-objective optimization in python. IEEE Access **8**, 89497–89509 (2020)

2. Cheng, R., Li, M., Tian, Y., et al.: A benchmark test suite for evolutionary many-objective optimization. Complex & Intelligent Systems **3**(1), 67–81 (2017)
3. Cococcioni, M., Pappalardo, M., Sergeyev, Y.: Lexicographic Multi-Objective Linear Programming using Grossone Methodology: Theory and Algorithm. Appl. Math. and Comp. **318**, 298–311 (2018)
4. Cococcioni, M., Fiaschi, L.: The Big-M method with the numerical infinite M. Optimization Letters (2020). https://doi.org/10.1007/s11590-020-01644-6
5. Cococcioni, M., et al.: Solving the Lexicographic Multi-Objective Mixed-Integer Linear Programming Problem using Branch-and-Bound and Grossone Methodology. Comm. in Nonlin. Sc. and Num. Sim. **84**, 105177 (2020)
6. De Cosmis, S., De Leone, R.: The use of Grossone in Mathematical Programming and Operations Research. Appl. Math. and Comp. **218**(16), 8029–8038 (2012)
7. De Leone, R.: Nonlinear programming and grossone: Quadratic programing and the role of constraint qualifications. Appl. Math. and Comp. **318**, 290 – 297 (2018)
8. De Leone, R., Fasano, G., Roma, M., Sergeyev, Y.D.: Iterative grossone-based computation of negative curvature directions in large-scale optimization. Journal of Optimization Theory and Applications **186**(2), 554–589 (Aug 2020)
9. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. IEEE Trans. Evol. Comp. **18**(4), 577–601 (2014)
10. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comp. **6**(2), 182–197 (2002)
11. Deb, K.: Multi-objective optimization using evolutionary algorithms. John Wiley & Sons (2001)
12. Gaur, A., Khaled Talukder, A., Deb, K., Tiwari, S., Xu, S., Jones, D.: Unconventional optimization for achieving well-informed design solutions for the automobile industry. Engin. Optimiz. pp. 1–19 (2019)
13. Huband, S., et al.: A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans. Evol. Comp. **10**(5), 477–506 (2006)
14. Lai, L., Fiaschi, L., Cococcioni, M.: Solving Mixed Pareto-Lexicographic Multi-Objective Optimization Problems: The Case of Priority Chains. Swarm and Evol. Comp. p. 100687 (2020)
15. Liao, X., Li, Q., Yang, X., Zhang, W., Li, W.: Multiobjective optimization for crash safety design of vehicles using stepwise regression model. Structural and multidisciplinary optimization **35**(6), 561–569 (2008)
16. Marler, R.T., Arora, J.S.: Survey of multi-objective optimization methods for engineering. Structural and multidisciplinary optimization **26**(6), 369–395 (2004)
17. Schmiedle, F., Drechsler, N., Große, D., Drechsler, R.: Priorities in multi-objective optimization for genetic programming. In: In Proc. of the 3rd Annual Conf. on Genetic and Evolutionary Comp. pp. 129–136. Morgan Kaufmann (2001)
18. Schutze, O., Esquivel, X., Lara, A., Coello, C.A.C.: Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization. IEEE Transactions on Evolutionary Computation **16**(4), 504–522 (2012)
19. Seada, H., Deb, K.: U-NSGA-III: a unified evolutionary optimization procedure for single, multiple, and many objectives: proof-of-principle results. In: International conference on evolutionary multi-criterion optimization. pp. 34–49. Springer (2015)
20. Sergeyev, Y.D.: Numerical infinities and infinitesimals: Methodology, applications, and repercussions on two Hilbert problems. EMS Surveys Math. Sci **4**(2), 219—-320 (2017)
21. Zhang, Q., Li, H.: MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans. on Evol. Comp. **11**(6), 712–731 (2007)