ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240977

Generating Fair Solutions of Minimal Cost

Mohammed Bachir Bederina[a], **Djamal Chaabane**[a] **and Thibaut Lust**[b,*]

[a]University of Sciences and Technology Houari Boumediene, AMCD-RO Laboratory, DGRSDT, Algiers, Algeria
[b]Sorbonne Université, CNRS, LIP6, Paris, France

Abstract. In this work, we consider combinatorial multi-agent optimization problems, i.e., problems presenting a combinatorial set of solutions, and each solution is evaluated through a vector. An element of the vector corresponds to the utility that an individual agent receives from the solution. Given potential conflicts, it is improbable that a single feasible solution will be optimal for all agents. Consequently, a relevant objective is to identify solutions that are fair to all agents. There are several approaches to defining fairness in the context of optimization problems, and here we focus on Lorenz-optimal solutions. However, in some cases, in addition to the search for a fair solution, an economic criterion comes into play, i.e., we also seek to find a least-cost solution. Since the optimal solution for the cost function is not necessarily fair (i.e., Lorenz-optimal in our case), our aim is to generate a Lorenz-optimal solution with minimum cost. We propose a new exact method to solve this problem, and apply it to the multi-agent assignment problem and to the multi-agent knapsack problem. Results show that the new method is much more efficient than a method based on a complete enumeration of Lorenz-optimal solutions.

1 Introduction

This work explores scenarios in which decisions must be made with the involvement of multiple agents, each possessing its own utility function across a range of potential choices. These scenarios are common in practical situations. Consider for instance multi-agent scheduling problems where each agent needs to perform a set of jobs on shared machines, each job demanding specified processing times. Each agent gives a utility value for each job done on time. A globally optimal solution, such as one that maximizes the total utility, may not necessarily be the best solution because it can result in unfair outcomes. Any solution should emerge from a decision-making process that includes all agents. The question then arises: how can we define and identify a good compromise solution?

One way of approaching this question is to integrate a criterion of equity into the proposed solutions. Recently, a large body of research has focused on defining what constitutes fair outcomes. This has been particularly studied in the context of resource allocation problems, where a decision-maker is tasked with distributing goods (divisible or indivisible) among several agents, to satisfy an appropriate equity criterion. Many fairness criteria have been studied in the literature (egalitarian criterion [7], Nash product [10], ordered weighted average [20], proportionality [30], envy-freeness [34], etc.), see the recent survey of Amanatidis et al. [2]. However, to our knowledge, incorporating an economic criterion has not been explored in the search for equitable solutions. Studies do exist on the search for trade-offs between fairness and efficiency of a solution, see e.g., Bertsimas et al. [6], but the problem considered here is different: a solution is evaluated by an auxiliary (linear) cost function, independent of the utility function. Our aim is to generate a minimum-cost solution. However, this solution is rarely fair. Let's consider, for example, the classic problem of assigning objects to agents, where each agent has preferences over the objects (modeled by utility functions). In general, the objective is to find a fair solution, i.e., one that satisfies all the agents. However, allocating an object to an agent may have a certain cost (e.g., the cost of transporting an object to an agent). In this case, a fair solution will not necessarily be of minimal cost, and the optimal solution for the cost function is not necessarily fair. The goal is thus to find a solution that achieves a good compromise between fairness and cost. We deal with this problem as follows: given that there are some degrees of freedom in the definition of fairness, we will use a dominance relation that integrate fairness: the Lorenz dominance. Then the problem comes down to finding a minimum-cost Lorenz-optimal solution. It has been widely studied in the context of Pareto dominance, i.e., the search for a solution that optimizes a linear function (the cost function) under the constraint that the solution is Pareto-optimal [1, 16, 32]. But to our knowledge, this problem has never been studied for Lorenz dominance.

The paper is organized as follows. In the next section, we present the Lorenz dominance in the context of multi-agent combinatorial optimization problems. In Section 3, we introduce the main problem studied in this paper: optimizing a cost function over the set of Lorenz-optimal solutions. In Section 4, we present the first contribution of this paper: a new general method for generating all Lorenz-optimal solutions of multi-agent optimization problems. In Section 5, the second contribution of this paper is developed, and consists in a new method specifically dedicated to the problem of finding the minimum-cost Lorenz-optimal solution. In Section 6, we report results for two multi-agent combinatorial optimization problems: the assignment and the knapsack problems.

2 Fairness and Lorenz Dominance

We define a combinatorial multi-agent optimization problem as follows. Given $\mathcal{X}$ a feasible set, defined by a set of linear constraints on n decision variables, a solution $x \in \mathcal{X}$ is evaluated through different linear utility functions, that associate to a solution x a vector $y(x) = (f_1(x), f_2(x), \dots, f_p(x)) \in \mathbb{N}^p$, where $f_i(x)$ corresponds to the utility that agent i has for solution x (a number p of agents is considered). We aim to maximize the utility functions:

* Corresponding Author. Email: thibaut.lust@lip6.fr.

$$\underset{x \in \mathcal{X}}{\text{maximize}}\big(f_1(x), \dots, f_p(x)\big)$$

As in general there does not exist a solution optimal for all utility functions, we need a dominance relation to compare solutions. However, using the Pareto dominance to discriminate between the solutions proves insufficient, since a Pareto-optimal solution might not be fair to all agents (an agent might be completely disadvantaged compared to the others), as shown in the following example.

Example 1. *The following table displays the utilities (score between 1 and 9) given by four agents (a_1, a_2, a_3 and a_4) to four items (i_1, i_2, i_3, i_4). We look for the best assignment possible of the items to the agents (an agent can receive only one item, and an item can be assigned to only one agent).*

	i_1	i_2	i_3	i_4
a_1	4	8	2	1
a_2	8	6	5	2
a_3	9	4	4	7
a_4	3	6	1	1

There are 24 assignments possible ($4!$). Let's consider that a feasible assignment is evaluated through a four-component vector u, where each component u_i corresponds to the utility of the item assigned to the agent a_i. Let's note a feasible assignment as a 4-tuple, where the value of the element at position i represents the index of the item assigned to agent a_i.
Among all the feasible assignments, there are 8 Pareto-optimal assignments, given in the following table.

Solution	*Evaluation* ($y(x)$)
$x_1 = (1,3,4,2)$	$(4,5,7,6)$
$x_2 = (2,1,4,3)$	$(8,8,7,1)$
$x_3 = (2,3,1,4)$	$(8,5,9,1)$
$x_4 = (2,3,4,1)$	$(8,5,7,3)$
$x_5 = (3,1,4,2)$	$(2,8,7,6)$
$x_6 = (3,2,1,4)$	$(2,6,9,1)$
$x_7 = (3,4,1,2)$	$(2,2,9,6)$
$x_8 = (4,3,1,2)$	$(1,5,9,6)$

However, among these solutions there are many solutions that are not fair: solution x_8 gives the worst item possible for agent a_1, same for solution x_7 for agent a_2, and same for solutions x_2, x_3 and x_6 for agent a_4. We can therefore see that it would be necessary to use a more restrictive relation that incorporate fairness.

Ensuring fairness between several agents reveals to be a tricky question. The notion of fairness has been widely investigated in economics and social choice and has lead to different definitions. In economics, some measures of inequality of outcome distributions have been proposed, as the *Gini index* [12] or the *Atkinson index* [3]. In welfare economics, one refers to social welfare functions to evaluate the relative goodness of the alternatives with respect to the individual utilities [27].

The classic *utilitarian* criterion consists in evaluating a solution according to the sum of the utilities of the agents. However, it is insensitive to the distribution of the total sum of the individual's utilities [28] and can provide unfair solutions since it allows compensations between strongly satisfied agents and poorly satisfied ones. For example, a solution whose utility vector equals $(200, 1)$ will be considered better than a solution whose utility vector is $(100, 100)$. The classic *egalitarian* criterion (or its lexicographical refinement) overcomes this drawback by evaluating a solution with respect to the utility of its least satisfied agent (which is equivalent to the *max min* criterion). However, by focusing only on the utility of one agent (the least satisfied), high quality solutions can be eliminated by this aggregation function. For example, a solution with a utility vector $(100, 100, 100, 100)$ will be considered better than a solution with a utility vector $(99, 200, 200, 200)$, while this last solution is much better for three of the agents and just a little worse for the first agent. Some other aggregation functions, such as the *Ordered Weighting Average (OWA)* [35], enable to favor solutions for which the utilities of the agents are well-balanced, but they require an additional preferential information (appropriate weights).

In this work, we use the Lorenz dominance to compare solutions. Lorenz dominance is an elegant refinement of Pareto dominance to incorporate fairness. It has been proposed in economics to measure the inequalities in income distributions. It refines the Pareto dominance by selecting only the better distributed solutions. Roughly speaking, the Lorenz dominance enables to select all Pareto-optimal solutions that realize well-balanced compromises between the utilities of the agents, while not eliminating high-performance solutions. It encompasses the utilitarian and the egalitarian criterion and has been used to characterize equitable solutions in multi-objective optimization [17, 18] and robust solutions in decision under uncertainty [25]. It has also been studied within the framework of convex-cone theory [36], in multi-objective programming [4], and in metaheuristics [11, 21].

The Lorenz dominance relies on the construction of particular vectors, called *generalized Lorenz vectors*, obtained as follows.

Definition 1. *The* generalized Lorenz vector *(or simply* Lorenz vector*) of $y \in \mathbb{R}^p$ is the vector $L(y) \in \mathbb{R}^p$ defined by: $L(y) = (y_{(1)}, y_{(1)} + y_{(2)}, \dots, y_{(1)} + y_{(2)} + \dots + y_{(p)})$, where $(y_{(1)}, y_{(2)}, \dots, y_{(p)})$ represents the components of y sorted from the worst to the best (i.e., $y_{(1)} \geq y_{(2)} \geq \dots \geq y_{(p)}$ in the case of minimization, and $y_{(1)} \leq y_{(2)} \leq \dots \leq y_{(p)}$ in the case of maximization).*

Definition 2. *Pareto dominance relation: we say that a point $u = (u_1, \dots, u_p) \in \mathbb{R}^p$* Pareto dominates *a point $v = (v_1, \dots, v_p) \in \mathbb{R}^p$ if, and only if, $u_k \geq v_k, \forall\, k \in \{1, \dots, p\} \wedge \exists\, k \in \{1, \dots, p\} : u_k > v_k$. We denote this relation by $u \succ_P v$.*

Definition 3. *Lorenz dominance relation: we say that a point $u = (u_1, \dots, u_p) \in \mathbb{R}^p$* Lorenz dominates *a point $v = (v_1, \dots, v_p) \in \mathbb{R}^p$, if, and only if $L(u) \succ_P L(v)$. We denote this relation by $u \succ_L v$.*

Definition 4. *Lorenz-optimal solution: a feasible solution $x^* \in \mathcal{X}$ is called* Lorenz-optimal *if there is no other feasible solution $x \in \mathcal{X}$ such that $y(x) \succ_L y(x^*)$.*

The *Lorenz-optimal set* denoted by X_L contains all Lorenz-optimal solutions.

Example 2. *In the following table, we indicate the Lorenz vector associated to each Pareto-optimal solution of the multi-agent assignment problem of Example 1. If we compare the Lorenz vectors with the Pareto dominance relation, we can easily check that only three are Pareto non-dominated (represented in bold). Therefore, only the corresponding solutions (x_1, x_2 and x_4) are Lorenz-optimal solutions. We see that the solution x_2 is the best for the sum of the utilities (the sum is given by the last component of the Lorenz vector). Solution x_1 is the best for the max min criterion (given by the first component of the Lorenz vector), while solution x_4 establishes a good compromise between these two criteria.*

Solution	*Evaluation* ($y(x)$)	*Lorenz vector* ($L(y(x))$)
$x_1 = (1, 3, 4, 2)$	$(4, 5, 7, 6)$	$(\mathbf{4}, \mathbf{9}, \mathbf{15}, \mathbf{22})$
$x_2 = (2, 1, 4, 3)$	$(8, 8, 7, 1)$	$(\mathbf{1}, \mathbf{8}, \mathbf{16}, \mathbf{24})$
$x_3 = (2, 3, 1, 4)$	$(8, 5, 9, 1)$	$(1, 6, 14, 23)$
$x_4 = (2, 3, 4, 1)$	$(8, 5, 7, 3)$	$(\mathbf{3}, \mathbf{8}, \mathbf{15}, \mathbf{23})$
$x_5 = (3, 1, 4, 2)$	$(2, 8, 7, 6)$	$(2, 8, 15, 23)$
$x_6 = (3, 2, 1, 4)$	$(2, 6, 9, 1)$	$(1, 3, 9, 18)$
$x_7 = (3, 4, 1, 2)$	$(2, 2, 9, 6)$	$(2, 4, 10, 19)$
$x_8 = (4, 3, 1, 2)$	$(1, 5, 9, 6)$	$(1, 6, 12, 21)$

We see thus through this example the interest of using the Lorenz dominance, since solutions presenting a good compromise between the utilitarian criterion and the egalitarian criterion are also generated.

The Lorenz dominance is closely related to the notion of *Pigou-Dalton transfers*. In social choice theory, a Pigou-Dalton transfer is an income transfer from a richer to a poorer person by an amount less than or equal to their initial income difference.

Definition 5. *Transfer principle [15]: let $y \in \mathbb{R}^p$ such that $y_i > y_j$ for some i, j. Then for all ε such that $0 \leq \varepsilon \leq y_i - y_j$, we have that $y - \varepsilon\, e_i + \varepsilon\, e_j \succsim_L y$ where e_i (resp. e_j) is the vector whose i^{th} (resp. j^{th}) component equals 1, all others being 0.*

This principle means that for some cost-vector $y \in \mathbb{R}^p$ with $y_i > y_j$, slightly increasing y_j and decreasing y_i while preserving the mean of the costs would produce a better distribution of the costs, and consequently a more balanced solution. For example, the distribution of the vector $y = (20, 20)$ is better than the distribution of the vector $y' = (10, 30)$, since it can be obtained from y' by a transfer of size 10. This principle enables to compare vectors with the same mean. Note that using a similar transfer of size greater than 20 would increase the inequality between utilities distribution. This explains why the transfers must have a size $\varepsilon \leq y_i - y_j$.

The generalized Lorenz extension that we consider here enables to compare vectors with different means thanks to the *Pareto-monotonicity* property [29], which means that if a vector y^1 Pareto dominates another vector y^2, then y^1 Lorenz dominates y^2.

Property 1. *Pareto-monotonicity: $\forall y^1, y^2 \in \mathbb{R}^p, y^1 \succ_P y^2 \Rightarrow y^1 \succ_L y^2$.*

If we look again at Example 1, we can now explain why, e.g., x_3 is Lorenz dominated by x_2. We have that $y(x_2) = (8, 8, 7, 1) \succ_P (8, 7, 7, 1)$. Therefore, $y(x_2) \succ_L (8, 7, 7, 1)$ by the Pareto-monotinicity principle. Furthermore, we have that $(8, 7, 7, 1) \succ_L y(x_3) = (8, 5, 9, 1)$ by the transfer principle (transfer of 2 from the third agent to the second agent). By transitivity, we have that $y(x_2) \succ_L y(x_3)$ and x_3 is thus Lorenz dominated by x_2.

3 Optimization of a Cost Function over the Set of Lorenz-Optimal Solutions

We present now the main problem studied in this paper. Let's consider an auxiliary linear cost function $\varphi : \mathcal{X} \to \mathbb{R}$, which evaluates a solution x according to an economic criterion. The optimization of a cost function over the set of Lorenz-optimal solutions amounts to the following problem:

$$\underset{x \in X_L}{\text{minimize}}\ \varphi(x)$$

i.e., we search for a solution of minimal cost among all Lorenz-optimal solutions.

The difficulty of this problem lies in the fact that the set X_L is not characterized, either explicitly by a set of solutions, or even implicitly by a set of constraints. Moreover, optimizing $\varphi(x)$ over the set of feasible solutions generally yields only a dominated Lorenz solution.

Example 3. *Let's go back to the multi-agent assignment problem of Example 1. We consider now that assigning an item to an agent has a cost. All assignment costs are given by the following matrix:*

	i_1	i_2	i_3	i_4
a_1	*6*	*5*	*3*	*2*
a_2	*5*	*4*	*1*	*5*
a_3	*5*	*4*	*3*	*5*
a_4	*5*	*6*	*2*	*6*

For example, assigning the first item to the first agent costs 6 units. It can be shown that the solution minimizing the total assignment cost is the solution $x_9 = (4, 3, 2, 1)$ with a total cost equal to 12 (this solution can be easily obtained in polynomial time with the Hungarian method [19]). However, this solution does not satisfy the agents, as the utility vector is equal to $(1, 5, 4, 3)$ and therefore x_9 is Pareto dominated by other solutions (which implies that x_9 is not Lorenz-optimal). We rather need to look for the best Lorenz-optimal solution for the cost function. We therefore evaluate all Lorenz-optimal solutions found previously (see Example 2) with the cost function and see that solution x_4 is the best, as shown in the following table.

Solution	*Evaluation*	*Cost*
$x_1 = (1, 3, 4, 2)$	$y(x_1) = (4, 5, 7, 6)$	$\varphi(x_1) = 18$
$x_2 = (2, 1, 4, 3)$	$y(x_2) = (8, 8, 7, 1)$	$\varphi(x_2) = 17$
$x_4 = (2, 3, 4, 1)$	$y(x_4) = (8, 5, 7, 3)$	$\varphi(x_4) = \mathbf{16}$

4 Generating all Lorenz-Optimal Solutions

A first method to solve the problem of optimizing a cost function over the set of Lorenz-optimal solutions is to enumerate the set X_L containing all Lorenz-optimal solutions and select from X_L a solution minimizing $\varphi(x)$.

The Lorenz-optimal solutions could be generated through a two-stage procedure that first generates all Pareto-optimal solutions and second selects only the Lorenz-optimal ones among them (since a Lorenz-optimal solution is necessarily a Pareto-optimal solution). However, the problems considered here could present a high number of agents (e.g., up to 100, see the experiments in Section 6) and therefore the size of the Pareto-optimal set will be too high to be enumerated (for such number of agents we almost have that every feasible solution is Pareto-optimal).

4.1 State-of-the-Art

To our knowledge, only a few works address the problem of Lorenz optimization for multi-agent problems. We briefly list below the methods proposed in the literature.

Ranking Method. The ranking method has been proposed by Perny et al. [25] in a robust optimization setting. This method works simply by computing the solutions in non-increasing order of their sum using a k-best algorithm. Indeed, the sum of the objective values correspond to the last component of the Lorenz vector. Therefore, a solution maximizing the sum is weakly Lorenz Pareto-optimal. It is then necessary to define a valid stopping criterion for the k-best algorithm. The stopping criterion is based on the following proposition: a vector $(y_1, \ldots, y_p)$ Lorenz dominates any vector $(y'_1, \ldots, y'_p)$

such that $\sum_{i=1}^{p} y'_i < p \cdot y_{(1)}$ where $y_{(1)} = \min(y_1, \ldots, y_p)$. Consequently, once the ranking algorithm generates a solution for which the stopping criterion is satisfied, the method can be stopped as all Lorenz-optimal solutions have been generated. This method can be used with any number of utility functions, but it has only been tested up to $p = 5$ on the robust shortest path problem.

Dynamic Programming Based Method. One cannot use directly a dynamic programming procedure to generate all Lorenz-optimal solutions to a multi-agent combinatorial problem (the Bellman principle does not hold for Lorenz dominance [24, 26]). However, since dynamic programming can be used with Pareto dominance, Perny and Spanjaard [24] have proposed to adapt a multi-objective dynamic programming based procedure to Lorenz optimization by adding a valid dominance rule. The method has been applied to find robust paths in state space graphs.

Two-Phase Method. Galand and Lust [14] have adapted the classic two-phase method for Pareto optimization [33] in the case of Lorenz dominance. The method is effective, has been tested on shortest path and set covering problems but it only works for $p = 2$ utility functions, while we will deal with much more number of agents.

4.2 New Method

The new method that we propose for enumerating all Lorenz-optimal solutions of multi-agent combinatorial optimization problems is based on the method of Sylva and Crema [31] for generating all Pareto-optimal solutions of multi-objective combinatorial optimization problems. The principle of this method is as follows: at each stage, a positive combination of all objective functions is optimized, with the addition of constraints stipulating that the generated solution must not be Pareto dominated by previously generated solutions (i.e., presenting an improvement on at least one of the objectives over all other previously generated solutions). By iterating this simple process, all Pareto-optimal can be generated. However, it does not work with Lorenz dominance since a solution optimizing a positive combination of all objective functions is not necessarily Lorenz-optimal (contrary to Pareto dominance).

We rather need to optimize Ordered Weighted Average (OWA) functions, as Ogryczak [23] shown that any solution optimizing an OWA function with strictly decreasing and strictly positive weights is Lorenz-optimal.

The OWA function has been introduced by Yager [35]:

Definition 6. *Given a vector $y \in \mathbb{R}^p$ and a weighting vector $w \in [0,1]^p$, the* ordered weighted average (OWA) *of y with respect to w is defined by: $f^{owa}(y, w) = \sum_{k=1}^{p} w_k y_{(k)}$ where $y_{(1)} \leq \ldots \leq y_{(p)}$.*

It can be shown (see e.g., [25]) that optimizing an OWA function with strictly decreasing and strictly positive weights for a multi-agent combinatorial optimization problem boils down to optimizing a positive linear combination of Lorenz vector, i.e., we have:

$$x \in \text{argmax } OWA(f(x), w) \Leftrightarrow x \in \text{argmax} \sum_{k=1}^{p} \lambda_k \, L_k(f(x))$$

with $w_i = \sum_{k=i}^{p} \lambda_k, \forall i \in \{1, \ldots, p\}$, and $\lambda_k > 0, \forall k \in \{1, \ldots, p\}$.

However, as Lorenz vectors are rank-dependent, it is not possible to directly use an Integer Linear Program (ILP) solver to obtain optimal solutions for OWA functions. It is necessary to linearize this operator. We have used the linearization proposed by Ogryczak [23]. It is based on the following linear program, allowing to retrieve the Lorenz vector from any vector $y \in \mathbb{R}^p$:

$$L_k(y) = \begin{cases} \min & \sum_{i=1}^{p} \alpha_i^k y_i \\ s.t. & \\ & \sum_{i=1}^{p} \alpha_i^k = k \\ & 0 \leq \alpha_i^k \leq 1 \quad i = 1, \ldots, p \end{cases}$$

However, it is more convenient to work with the dual (since we will need to maximize a linear combination of the Lorenz vectors) given below:

$$L_k(y) = \begin{cases} \max & k \times r_k - \sum_{i=1}^{p} b_i^k \\ s.t. & \\ & r_k - b_i^k \leq y_i \quad i = 1, \ldots, p \\ & b_i^k \geq 0 \quad i = 1, \ldots, p \end{cases}$$

We can then obtain a first Lorenz-optimal solution of any multi-agent combinatorial problems with p utility functions $(f_1(x), \ldots, f_p(x))$ and a feasible set $\mathcal{X}$ by optimizing a positive linear combination of Lorenz vector using the following ILP with any strictly decreasing and strictly positive weights ($w_1 > w_2 > \cdots > w_p > 0$):

$$P_1 \begin{cases} \max & \sum_{k=1}^{p} w_k (k \times r_k - \sum_{i=1}^{p} b_i^k) \\ s.t. & \\ & r_k - b_i^k \leq f_i(x) \quad i, k = 1, \ldots, p \\ & b_i^k \geq 0 \quad i, k = 1, \ldots, p \\ & x \in \mathcal{X} \end{cases}$$

Then the remaining Lorenz-optimal solutions can be obtained by solving the same model, with the additional constraint that the new solution generated should not be Lorenz dominated by any of the previous solutions:

$$P_L \begin{cases} \max & \sum_{k=1}^{p} w_k (k \times r_k - \sum_{i=1}^{p} b_i^k) \\ s.t. & \\ & r_k - b_i^k \leq f_i(x) \quad i, k = 1, \ldots, p \\ & k \times r_k - \sum_{i=1}^{p} b_i^k \geq (L_k^s + 1) z_k^s \quad k = 1, \ldots, p; \\ & \quad s = 1, \ldots, l \\ & \sum_{k=1}^{p} z_k^s \geq 1 \quad s = 1, \ldots, l \\ & b_i^k \geq 0 \quad i, k = 1, \ldots, p \\ & z_k^s \in \{0, 1\} \quad k = 1, \ldots, p; \\ & \quad s = 1, \ldots, l \\ & x \in \mathcal{X} \end{cases}$$

Where l is the number of Lorenz-optimal solutions generated so far, L_k^s is component k of the Lorenz vector of a Lorenz-optimal solution x^s, and z^s is a binary vector of size p, associated to each solution x^s, allowing to impose that the new solution generated is better than all previously generated Lorenz-optimal solutions for at least one component of the Lorenz vector. By solving this ILP, either a new Lorenz-optimal solution is generated, or the problem is infeasible, which means that all Lorenz-optimal solutions have been generated.

Note that we can have solutions with the same Lorenz vector, but with different utility vectors. For example, for $y^1 = (7, 4, 6)$ and $y^2 = (4, 6, 7)$, we have $L(y^1) = L(y^2) = (4, 10, 17)$. Once we have generated a Lorenz-optimal solution, we then need to solve the following ILP to possibly generate new solutions with the same Lorenz vector:

$$P_s \begin{cases} \max & \sum_{k=1}^{p} w_k(k \times r_k - \sum_{i=1}^{p} b_i^k) \\ s.t. & \\ & r_k - b_i^k \leq f_i(x) \quad i,k = 1,\dots,p \\ & k \times r_k - \sum_{i=1}^{p} b_i^k = L_k \quad k = 1,\dots,p \\ & f_i(x) \geq (y_i^s + 1)z_i^s \quad i = 1,\dots,p; \; s = 1,\dots,l \\ & \sum_{i=1}^{p} z_i^s \geq 1 \quad s = 1,\dots,l \\ & b_i^k \geq 0 \quad i,k = 1,\dots,p \\ & z_i^s \in \{0,1\} \quad i = 1,\dots,p; \; s = 1,\dots,l \\ & x \in \mathcal{X} \end{cases}$$

Where l is the number of Lorenz-optimal solutions generated so far, L_k is component k of the Lorenz vector of the previously generated Lorenz-optimal solution, and z^s is a binary vector of size p, associated to each solution x^s, allowing to impose that the new solution generated is better than all previously generated Lorenz-optimal solutions for at least one component of the utility vector.

By solving iteratively P_L and P_s, we can generate all Lorenz-optimal solutions (see Algorithm 1).

Algorithm 1: Generate the Lorenz-optimal set X_L

Data: $\mathcal{X}, f_i(x), i = 1,\dots,p$
Result: X_L
Solve P^1 to get a first solution x^1;
$X_L \leftarrow \{x^1\}$;
Solve P_s with $L(x^1)$ as Lorenz vector;
while P_s *is feasible* **do**
  $x^s \leftarrow$ solution returned by P_s;
  $X_L \leftarrow X_L \cup \{x^s\}$;
  Solve P_s with $L(x^1)$ as Lorenz vector;
end
Solve P_L;
while P_L *is feasible* **do**
  $x^s \leftarrow$ solution returned by P_L;
  $X_L \leftarrow X_L \cup \{x^s\}$;
  Solve P_s with $L(x^s)$ as Lorenz vector;
  while P_s *is feasible* **do**
    $x^s \leftarrow$ solution returned by P_s;
    $X_L \leftarrow X_L \cup \{x^s\}$;
    Solve P_s with $L(x^s)$ as Lorenz vector;
  end
  Solve P_L;
end
Return X_L

Example 4. *Let's go back to the multi-agent assignment problem of Example 3. By applying Algorithm 1, we can generate all Lorenz-optimal solutions and then select the one of minimal cost. Let's take* $w = (1, 0.99, 0.98, 0.97)$. *Solving* P_1 *gives the first Lorenz-optimal solution* $x_1 = (2, 1, 4, 3)$*, with* $y(x_1) = (8, 8, 7, 1)$ *and* $L(y(x_1)) = (1, 8, 16, 24)$. P_s *is unfeasible, showing that there are no other solutions sharing the same Lorenz vector. Solving* P_L *then gives* $x_2 = (2, 3, 4, 1)$*, with* $y(x_2) = (8, 5, 7, 3)$ *and* $L(y(x_2)) = (3, 8, 15, 23)$. P_s *is once again unfeasible. Finally, solving* P_L *gives the solution* $x_3 = (1, 3, 4, 2)$*, with* $y(x_3) = (4, 5, 7, 6)$ *and* $L(y(x_3)) = (4, 9, 15, 22)$. *All Lorenz-optimal solutions have been found, the problem* P_L *is unfeasible and the algorithm stops.*

5 Generating a Lorenz-Optimal Solution of Minimal Cost

The set of Lorenz-optimal solutions can be very large (many multi-agent combinatorial optimization problems have been proved to be intractable[1], see e.g., [14, 25, 26]) and, in order to find the lowest-cost Lorenz-optimal solution, it is more interesting to develop methods that only partially enumerate X_L. We have therefore adapted the method developed by Jorge [16] in the context of Pareto dominance to Lorenz dominance. The method that we propose is based on a Lorenz efficiency test and on the addition of constraints specific to Lorenz dominance in order to limit the size of the search space.

The method starts by calculating a lower bound on the value of the minimum-cost Lorenz-optimal. This lower bound v^l is obtained by solving the following relaxed problem P_r, where the cost function $\varphi(x)$ is optimized over the feasible set $\mathcal{X}$ (and not over $\mathcal{X}_L$ which is unknown).

$$P_r \begin{cases} \min & \varphi(x) \\ s.t. & \\ & x \in \mathcal{X} \end{cases}$$

The solution of the relaxed problem P_r gives a solution $x^r \in \mathcal{X}$ with the best possible cost value. But, in general, this solution is not Lorenz-optimal. In order to test if a solution is Lorenz-optimal, we have developed a new Lorenz efficiency test, based on the test of Ecker and Kouada [13] for Pareto dominance. This efficiency test is performed by the following ILP, called E_L:

$$E_L \begin{cases} \max & \sum_{k=1}^{p} \phi_k \\ s.t. & \\ & k \times r_k - \sum_{i=1}^{p} b_i^k - \phi_k = L_k \quad k = 1,\dots,p \\ & r_k - b_i^k \leq f_i(x) \quad i,k = 1,\dots,p \\ & b_i^k \geq 0 \quad i,k = 1,\dots,p \\ & \phi_k \geq 0 \quad k = 1,\dots,p \\ & x \in \mathcal{X} \end{cases}$$

where L_k is the component k of the Lorenz vector of the solution to be tested for Lorenz efficiency. This efficiency test works

[1] A problem is intractable if it presents an exponential number of optimal solutions.

as follows: we try to find a solution $x \in \mathcal{X}$ dominating a solution presenting a Lorenz vector L, by looking for positive values $\phi_k, \forall k \in \{1, \dots, p\}$, such that each component value of the Lorenz vector of x is greater than or equal to the component value of the Lorenz vector of the solution to be tested (with a least one strict positive inequality). If such solution x exists, the optimal value of E_L, equal to $\sum_{k=1}^{p} \phi_k$, is then strictly positive and the solution x returned by E_L is a new Lorenz-optimal solution. Otherwise, the solution associated to the Lorenz vector L is Lorenz-optimal.

If x^r is not Lorenz-optimal, we try then to generate a solution minimizing the cost function $\varphi(x)$ and not Lorenz dominated by the Lorenz-optimal solutions generated previously. This is done with the following ILP, called P_L^{φ}, which is very similar to P_L, except that now we try to minimize the cost function $\varphi(x)$.

$$P_L^{\varphi}\begin{cases} \min & \varphi(x) \\ s.t. & \\ & r_k - b_i^k \leq f_i(x) \quad i,k = 1,\dots,p \\ & k \times r_k - \sum_{i=1}^{p} b_i^k \geq (L_k^s + 1) z_k^s \quad k = 1,\dots,p; \; s = 1,\dots,l \\ & \sum_{k=1}^{p} z_k^s \geq 1 \quad s = 1,\dots,l \\ & b_i^k \geq 0 \quad i,k = 1,\dots,p \\ & z_k^s \in \{0,1\} \quad k = 1,\dots,p; \; s = 1,\dots,l \\ & \varphi(x) < v^u \\ & \varphi(x) \geq v^l \\ & x \in \mathcal{X} \end{cases}$$

where φ^u and φ^l are bounds applied to the cost function $\varphi(x)$ to reduce the search space. If the problem P_L^{φ} is feasible, it yields a new solution that is not Lorenz dominated by the Lorenz-optimal solutions found so far. However, the solution returned by P_L^{φ} is not necessarily Lorenz-optimal (since the solution is only compared to a subset of X_L). A Lorenz efficiency test is then performed by solving E_L. Also, as stated in Section 4.2, we can have different solutions with the same Lorenz vector value. Therefore, after each Lorenz-optimal solution generated, we look if there exists solutions with the same Lorenz vector, but with a better value for the cost function $\varphi(x)$, which is done by solving P_s^{φ}, given below:

$$P_s^{\varphi}\begin{cases} \min & \varphi(x) \\ s.t. & \\ & k \times r_k - \sum_{i=1}^{p} b_i^k = L_k \quad k = 1,\dots,p \\ & r_k - b_i^k \leq f_i(x) \quad i,k = 1,\dots,p \\ & b_i^k \geq 0 \quad i,k = 1,\dots,p \\ & x \in \mathcal{X} \end{cases}$$

In short, the algorithm, given in details in Algorithm 2, alternates between the search for a minimum-cost solution, not Lorenz dominated by previously generated solutions, and Lorenz efficiency tests.

Example 5. *Let's go back to the multi-agent assignment problem of Example 3. We recall the utility and cost matrices associated to this problem.*

$$U = \begin{pmatrix} 4 & 8 & 2 & 1 \\ 8 & 6 & 5 & 2 \\ 9 & 4 & 4 & 7 \\ 3 & 6 & 1 & 1 \end{pmatrix} \qquad C = \begin{pmatrix} 6 & 5 & 3 & 2 \\ 5 & 4 & 1 & 5 \\ 5 & 4 & 3 & 5 \\ 5 & 6 & 2 & 6 \end{pmatrix}$$

The relaxed problem P_r is first solved, giving the solution $x^r = (4,3,2,1)$, with $y(x^r) = (1,5,4,3)$, $L(y(x^r)) = (1,4,8,13)$ and $\varphi(x^r) = 12$. We set the lower bound v^l to 12. We test the Lorenz efficiency of this solution with E_L. The test returns that x^r is Lorenz dominated by the solution $x^{s\prime} = (1,3,4,2)$, with $y(x^{s\prime}) = (4,5,7,6)$, $L(y(x^{s\prime}) = (4,9,15,22)$ and $\varphi(x^{s\prime}) = 18$. No solution with a similar Lorenz vector and a lower cost is found when P_s^{φ} is solved, thus $v^u = 18$, and the best solution found so far is $x^{s\prime}$. Then, the problem P_L^{φ} is solved, returning a new solution $x^t = (2,3,4,1)$, with $y(x^t) = (8,5,7,3)$, $L(y(x^t)) = (3,8,15,23)$, and $\varphi(x^t) = 16$. The Lorenz efficiency test confirms that this solution is Lorenz-optimal, the algorithm stops and returns the optimal solution $x^t = (2,3,4,1)$.

Algorithm 2: Generate a Lorenz-optimal solution of min cost

```
Data: φ(x), 𝒳, f_i(x), i = 1,...,p
Result: x* = argmin_{x∈X_L} φ(x)
Solve P_r to get the solution x^r;
v^l ← φ(x^r) ;
Test the Lorenz efficiency of x^r by solving E_L;
if x^r is Lorenz-optimal then
 |  Return x^r
end
x^s' ← solution returned by E_L;
Solve P_s^φ to find if there is a solution better than x^s' for φ(x);
x^s ← solution returned by P_s^φ;
x* ← x^s, v^u ← φ(x^s);
while v^u > v^l do
    Solve P_L^φ;
    if P_L^φ is infeasible then
     |  Return x* ;
    end
    else
        x^t ← solution returned by P_L^φ;
        if φ(x^t) > v^l then
         |  v^l ← φ(x^t) ;
        end
        Test the Lorenz efficiency of x^t by solving E_L;
        if x^t is Lorenz-optimal then
         |  Return x^t
        end
        x^s' ← solution returned by E_L;
        Solve P_s^φ to find if there is a solution better than x^s'
          for φ(x);
        x^s ← solution returned by P_s^φ;
        if φ(x^s) < v^u then
            v^u ← φ(x^s) ;
            x* ← x^s;
        end
    end
end
Return x*
```

6 Numerical Experiments

We have tested the two algorithms on two multi-agent combinatorial optimization problems: the Multi-Agent Assignment Problem (MAAP, same problem as in the different examples) and the Multi-Agent Knapsack Problem (MAKP). The MAKP is formulated as follows:

$$MAKP \begin{cases} \max f_k(x) = & \sum_{i=1}^{n} u_i^k x_i & k = 1, \ldots, p \\ \quad \text{s.t.} & & \\ & \sum_{i=1}^{n} w_i x_i \leq W & \\ & x_i \in \{0,1\} & i = 1, \ldots, n \end{cases}$$

Where n is the number of items, p is the number of agents, $u_i = (u_i^1, \ldots, u_i^p)$ is a utility vector associated to each item i, w_i is the weight of the item i and W is the weight capacity of the knapsack. The auxiliary cost function is given by $\varphi(x) = \sum_{i=1}^{n} c_i x_i$ where c_i is the cost associated to the item i. The goal is to find a Lorenz-optimal subset of items of minimal cost.

Computational experiments were carried out on an Intel i5-12500H processor. Integer linear programs were solved using CPLEX 22.1.1 under C++.

For the MAAP, we varied the number of agents p from 5 to 100, with different ranges of values for the utility and cost matrices (see Table 1). For the MAKP, we varied the number of items n from 5 to 100, with a number of agents p varying from 3 to 60, with different ranges of values for the utility matrix and the cost vector (see Table 2). For each algorithm, we report the computation time (in seconds) and the number of Lorenz-optimal solutions generated (Algorithm 1 generates all of them and then select the solution of minimal cost while Algorithm 2 generates only a subset of all Lorenz-optimal solutions before finding the one of minimal cost). Results are averaged over 20 instances.

As expected, the number of Lorenz-optimal solutions generated with Algorithm 2 is lower than the total number of Lorenz-optimal solutions generated with Algorithm 1. In addition, the time needed to calculate the minimum-cost Lorenz-optimal solution is significantly less with Algorithm 2, especially for larger instances. We also note that the higher the number of Lorenz-optimal solutions, the greater the difference in computation time between the two methods. The '-' symbol in the tables indicate cases where problems could not be solved in less than 10 minutes on average. Note that we have also tested the ranking method proposed by Perny et al. [25] to generate all Lorenz-optimal solutions of these problems, but the method failed to solve the considered instances within a reasonable computation time (under 10 minutes). This is mainly due to the high number of agents considered in the instances.

		Algorithm 2		Algorithm 1	
Range	Number of agents p	Time (s)	L-opt. sol. generated	Time (s)	Number of L-opt. sol.
[1,10]	5	0.15	1.1	0.14	1.5
	15	0.49	1.3	0.37	3.1
	20	0.29	1.0	0.60	3.2
	30	0.99	1.0	4.03	13.1
	40	3.42	1.0	12.84	12.4
	60	43.36	1.0	–	–
	100	329.62	1.0	–	–
[1,20]	5	0.13	1.1	0.13	1.4
	15	0.58	1.2	0.55	2.2
	20	0.99	1.2	1.51	2.3
	30	49.79	1.8	56.89	5.0
	40	60.86	1.4	330.89	15.7
[1,40]	5	0.11	1.0	0.18	1.6
	15	1.08	1.5	1.13	3.0
	20	5.49	1.7	16.87	3.3
[1,60]	5	0.15	1.2	0.16	1.4
	15	2.20	2.0	3.90	4.3
	20	22.71	2.1	346.87	5.2
[1,100]	5	0.13	1.1	0.14	1.6
	15	0.36	1.1	1.64	4.4
	20	2.65	1.6	62.50	4.8

Table 1. Results obtained for multi-agent assignment problem.

			Algorithm 2		Algorithm 1	
Range	n	Number of agents p	Time (s)	L-opt. sol. generated	Time (s)	Number of L-opt. sol.
[1,10]	5	10	0.20	1.3	0.24	2.5
		20	0.20	1.2	0.35	2.9
		60	1.06	1.3	3.37	3.1
	15	10	0.27	2.1	0.59	5.4
		20	0.48	2.0	1.33	5.8
		60	4.97	2.4	–	–
	40	5	0.52	4.1	2.91	17.0
		10	1.32	4.7	89.14	30.6
		20	4.34	5.9	–	–
	100	3	0.89	5.0	2.33	12.2
		5	10.15	10.9	–	–
[1,60]	5	20	0.23	3.5	0.29	2.5
		60	1.08	1.3	3.62	2.7
		100	6.85	1.3	–	–
	15	20	0.56	2.4	4.14	8.2
		40	1.47	2.2	–	–
	40	10	1.45	4.7	153.84	35.9
		20	6.97	6.2	–	–
	100	3	1.41	7.2	224.89	35.0

Table 2. Results obtained for the multi-agent knapsack problem.

7 Conclusion and Perspectives

In this paper, we have studied the addition of an economic function in multi-agent combinatorial optimization. The goal is to find a fair solution, considered here as a Lorenz-optimal solution, of minimal cost. Surprisingly, this problem had not yet been studied. We have presented two new approaches, based on existing methods developed for Pareto dominance, for solving this problem. The first approach is based on a complete enumeration of the Lorenz-optimal solutions, while the second approach directly seeks to optimize the cost function in order to limit the enumeration of all Lorenz-optimal solutions. The results on two multi-agents problems (assignment and knapsack) show the efficiency of the second method compared to the complete enumeration of Lorenz-optimal solutions. In the future, in order to be able to solve larger instances, it would be interesting to use specific search zones to reduce the number of constraints of the integer linear programs to be solved [8, 22, 32]. Another perspective is to integrate agents into the decision-making process, and learn their preferences while solving the problem, as has already been done in multi-objective combinatorial optimization [5, 9].

References

[1] M. Abbas and D. Chaabane. Optimizing a linear function over an integer efficient set. *European Journal of Operational Research*, 174(2): 1140–1161, 2006.

[2] G. Amanatidis, H. Aziz, G. Birmpas, A. Filos-Ratsikas, B. Li, H. Moulin, A. A. Voudouris, and X. Wu. Fair division of indivisible goods: Recent progress and open questions. *Artificial Intelligence*, 322: 103965, 2023.

[3] A. Atkinson. On the measurement of inequality. *Journal of Economic Theory*, 2(3):244–263, 1970.

[4] D. Baatar and M. Wiecek. Advancing equitability in multiobjective programming. *Computers & Mathematics with Applications*, 52:225–234, 2006.

[5] N. Benabbou, C. Leroy, and T. Lust. Regret-based elicitation for solving multi-objective knapsack problems with rank-dependent aggregators. In *24th European Conference on Artificial Intelligence, ECAI*, pages 419–426, 2020.

[6] D. Bertsimas, V. Farias, and N. Trichakis. On the efficiency-fairness trade-off. *Management Science*, 58(12):2234–2250, 2012.

[7] I. Bezáková and V. Dani. Allocating indivisible goods. *ACM SIGecom Exchanges*, 5(3):11–18, 2005.

[8] N. Boland, H. Charkhgard, and M. Savelsbergh. A new method for optimizing a linear function over the efficient set of a multiobjective integer program. *European Journal of Operational Research*, 260(3): 904–919, 2017.

[9] N. Bourdache and P. Perny. Active preference learning based on generalized gini functions: Application to the multiagent knapsack problem. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pages 7741–7748. AAAI Press, 2019.

[10] I. Caragiannis, D. Kurokawa, H. Moulin, A. D. Procaccia, N. Shah, and J. Wang. The unreasonable fairness of maximum nash welfare. *ACM Trans. Econ. Comput.*, 7(3), sep 2019.

[11] B. Chabane, M. Basseur, and J.-K. Hao. Lorenz dominance based algorithms to solve a practical multiobjective problem. *Computers & Operations Research*, 104:1–14, 2019.

[12] Y. Dodge. *Gini Index*, pages 231–233. Springer New York, 2008.

[13] J. G. Ecker and I. Kouada. Finding efficient points for linear multiple objective programs. *Mathematical Programming*, 8(1):375–377, 1975.

[14] L. Galand and T. Lust. Exact methods for computing all Lorenz optimal solutions to biobjective problems. In *Algorithmic Decision Theory, 4th International Conference, ADT*, volume 9346 of *Lecture Notes in Computer Science*, pages 305–321. Springer, 2015.

[15] G. Hardy, J. Littlewood, G. Pólya, et al. *Inequalities*. Cambridge university press, 1952.

[16] J. Jorge. An algorithm for optimizing a linear function over an integer efficient set. *European Journal of Operational Research*, 195:98–103, 2009.

[17] M. Kostreva and W. Ogryczak. Linear optimization with multiple equitable criteria. *RAIRO Operations Research*, 33:275–297, 1999.

[18] M. Kostreva, W. Ogryczak, and A. Wierzbicki. Equitable aggregations and multiple criteria analysis. *European Journal of Operational Research*, 158(2):362–377, 2004. Methodological Foundations of Multi-Criteria Decision Making.

[19] H. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.

[20] J. Lesca, M. Minoux, and P. Perny. The fair OWA one-to-one assignment problem: NP-hardness and polynomial time special cases. *Algorithmica*, 81(1):98–123, 2019.

[21] X. Li, H. Chehade, F. Yalaoui, and L. Amodeo. Lorenz dominance based metaheuristic to solve a hybrid flowshop scheduling problem with sequence dependent setup times. In *2011 International Conference on Communications, Computing and Control Applications (CCCA)*, pages 1–6, 2011.

[22] B. Lokman. Optimizing a linear function over the nondominated set of multiobjective integer programs. *International Transactions in Operational Research*, 28:2248–2267, 2019.

[23] W. Ogryczak. Inequality measures and equitable approaches to location problems. *European Journal of Operational Research*, 122(2):374–391, 2000.

[24] P. Perny and O. Spanjaard. An axiomatic approach to robustness in search problems with multiple scenarios. In *Proceedings of the 19th conference on Uncertainty in Artificial Intelligence, UAI*, pages 469–476, 2003.

[25] P. Perny, O. Spanjaard, and L.-X. Storme. A decision-theoretic approach to robust optimization in multivalued graphs. *Annals of Operations Research*, 147(1):317–341, 2006.

[26] P. Perny, P. Weng, J. Goldsmith, and J. Hanna. Approximation of Lorenz-optimal solutions in multiobjective markov decision processes. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI*, page 508–517. AUAI Press, 2013.

[27] A. Sen. Utilitarianism and welfarism. *Journal of Philosophy*, 76(9): 463–489, 1979.

[28] A. Sen. *On economic inequality*. Clarendon Press, expanded ed., 1997.

[29] A. Shorrocks. Ranking income distributions. *Economica*, 50(197):3–17, 1983.

[30] H. Steinhaus. Sur la division pragmatique. *Econometrica*, 17:315–319, 1949.

[31] J. Sylva and A. Crema. A method for finding the set of non-dominated vectors for multiple objective integer linear programs. *European Journal of Operational Research*, 158(1):46–55, 2004.

[32] S. Tamby and D. Vanderpooten. Optimizing over the efficient set of a multi-objective discrete optimization problem. In *21st International Symposium on Experimental Algorithms, SEA*, volume 265, pages 9:1–9:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[33] E. Ulungu and J. Teghem. The two-phases method: An efficient procedure to solve biobjective combinatorial optimization problems. *Foundation of Computing and Decision Science*, 20:149–156, 1995.

[34] H. R. Varian. Equity, envy, and efficiency. *Journal of Economic Theory*, 9(1):63–91, 1974.

[35] R. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man, and Cybernetics*, 18(1):183–190, 1988.

[36] P. Yu. Cone convexity, cone extreme points and nondominated solutions in decision problems with multiobjectives. *Journal of Optimization Theory and Applications*, 14:319–377, 1974.