Designing sustainable diet plans by solving triobjective integer programs

Luca Benvenuti, Alberto De Santis, Marianna De Santis, Daniele Patria

September 17, 2024

Abstract

We present an algorithm for triobjective nonlinear integer programs that combines the ε-constraint method with available oracles for biobjective integer programs. We prove that our method is able to detect the nondominated set within a finite number of iterations. Specific strategies to avoid the detection of weakly nondominated points are devised. The method is then used to determine the nondominated solutions of triobjective 0-1 models, built to design nutritionally adequate and healthy diet plans, minimizing their environmental impact. The diet plans refer to menus for school cafeterias and we consider the carbon, water and nitrogen footprints as conflicting objectives to be minimized. Energy and nutrient contents are constrained in suitable ranges suggested by the dietary recommendation of health authorities. Results obtained on two models and on real world data are reported and discussed.

1 Introduction

Multiobjective optimization is used by decision makers when considering more than one conflicting objective function simultaneously. Multiobjective optimization is applied to several fields such as chemical engineering, energy management, drug design [29, 32, 22], just to name a few. Some applications may require to model the problem using integer variables, see e.g. [17, 19, 26]. In these cases, we are in the context of multiobjective (mixed) integer programming (MOIP). It is the purpose of this paper to define and solve triobjective integer models to design nutritionally adequate and sustainable diet plans. In particular, we look for diet plans for school cafeterias that simultaneously minimize their carbon, water and nitrogen footprints. When dealing with more than one objective function, one looks for the so called nondominated solutions, points for which none of the objective functions can be improved without degrading some of the other objective values. In some applications, like ours, it is desirable to find the complete set of nondominated solutions so that the decision maker is able to choose among them. In this case, an exact algorithm is needed, i.e. an algorithm able to find the whole set of nondominated points of a MOIP.

There is a growing interest on exact algorithms for multiobjective mixed integer optimization and we mention the recent survey from Halffmann et al. [16] for a comprehensive overview on solution approaches for multiobjective mixed integer linear problems. Exact methods for nonlinear integer problems also exist and there is a distinction between those that work in the space of the decision variables (see e.g [5, 6, 7, 11, 21]), and those that work in the space of the objective functions (see e.g. [9, 8, 31]). This work presents an exact algorithm for triobjective integer programming problems of the following form

$$\begin{aligned} \min \quad & (f_1(x), f_2(x), f_3(x))^T \\ \text{s.t.} \quad & x \in \mathcal{X} \cap \mathbb{Z}^n, \end{aligned} \tag{TOIP}$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ and $f_1, f_2, f_3 : \mathbb{R}^n \to \mathbb{R}$ are continuous functions. Nonlinear functions can be handled by our approach, as long as they satisfy the so called positive γ property introduced in [9] that will be recalled later. The image of the feasible set $\mathcal{X} \cap \mathbb{Z}^n$ under the vector-valued function $f : \mathbb{R}^n \to \mathbb{R}^3$ represents the feasible set in the *criterion space*, or the *image set*. The *efficient solutions* of problem (TOIP) are points $x^* \in \mathcal{X} \cap \mathbb{Z}^n$ such that there exists no other feasible point $x \in \mathcal{X} \cap \mathbb{Z}^n$ for which $f_j(x) \leq f_j(x^*)$, $j = 1, 2, 3$ and $f(x) \neq f(x^*)$. The images $f(x)$ of efficient points $x \in \mathcal{X} \cap \mathbb{Z}^n$ are called *nondominated points*. Furthermore, a point $\bar{x} \in \mathcal{X} \cap \mathbb{Z}^n$ is called a *weakly efficient* point of (TOIP) if there is no $x \in \mathcal{X} \cap \mathbb{Z}^n$ with $f(x) < f(\bar{x})$, where $<$ is meant componentwise. The images $f(x)$ of weakly efficient solutions $x \in \mathcal{X} \cap \mathbb{Z}^n$ are called *weakly non-dominated points*. In the following, we will denote the set of non-dominated points of (TOIP), also called the *non-dominated set*, by $\mathcal{Y}_N$ and the set of weakly non-dominated points by $\mathcal{Y}_{wN}$.

The paper is organized as follows. In Section 2, we present our method for triobjective integer nonlinear problems. We analyze its correctness and present a strategy to avoid the detection of weakly nondominated points. A comparison on linear instances with two solvers for triobjective integer linear programming problems is reported in the Appendix. In Section 3, we present our application and two triobjective integer programs modeling the design of sustainable diet plans. We finally discuss the results obtained by solving the models using our algorithm. In Section 4, we draw some conclusions.

2 Algorithm `TrIntOpt`

The algorithm we propose extends the ideas used in [8] in order to define an exact criterion space method for triobjective nonlinear integer programs. Our algorithm, named `TrIntOpt`, is based on the ε-constraint method, a well-known scalarization technique. The idea is to iteratively solve biobjective subproblems, defined by adding further constraints to the original feasible set. More precisely, given (TOIP), at every iteration k our method determines the nondominated set of biobjective problems of the following form:

$$\begin{aligned} \min \quad & (f_1(x), f_2(x))^\top \\ \text{s.t.} \quad & f_3(x) \le \varepsilon^k \\ & x \in \mathcal{X} \cap \mathbb{Z}^n, \end{aligned} \qquad (\text{BOIP}^k)$$

where the parameter ε^k varies between $\min_{x\in\mathcal{X}\cap\mathbb{Z}^n} f_3(x)$ and the value $f_3(\hat{x}^0)-\delta$, being δ a positive step size and $\hat{x}^0$ the point defined as follows. Among the efficient points of the biobjective problem $\min_{x\in\mathcal{X}\cap\mathbb{Z}^n}(f_1(x), f_2(x))^\top$, $\hat{x}^0$ is one that achieves the maximum with respect to the objective function f_3. As it will be clarified later on, the step size δ controls the exacteness of `TrIntOpt`. The role of f_1, f_2 and f_3 in the definition of Problem (BOIP^k) can be interchanged.

For the definition of `TrIntOpt` we need to have an oracle able to detect the nondominated set of the biobjective nonlinear integer problem (BOIP^k):

Assumption 2.1. *There exists an oracle able to detect the complete nondominated set of Problem* (BOIP^k) *after having addressed a finite number B^k of single-objective integer programs. For each nondominated point $y \in \mathbb{R}^2$ detected, the oracle is able to compute one of its preimage, namely one efficient point $x \in \mathcal{X} \cap \mathbb{Z}^n$ such that $(f_1(x), f_2(x)) = y$.*

In the following, we denote by $\mathcal{E}^k$ the set of efficient points detected by the oracle in Assumption 2.1. We cite [8, 9] as works where algorithms satisfying Assumption 2.1 are defined. Furthermore, we need to assume the existence of the ideal point, in order to be guaranteed that the nondominated set is a finite set:

Assumption 2.2. *We assume that the ideal objective values $f_i^{\mathrm{id}} := \min_{\mathcal{X}\cap\mathbb{Z}^n} f_i(x)$, $i = 1, 2, 3$, and thus the ideal point $f^{\mathrm{id}} := (f_1^{\mathrm{id}}, f_2^{\mathrm{id}}, f_3^{\mathrm{id}}) \in \mathbb{R}^3$, exists.*

We report in Algorithm 1 the scheme of our method `TrIntOpt`. `TrIntOpt` starts by computing $x^* \in \mathcal{X} \cap \mathbb{Z}^n$ as the minimum with respect to f_3 and $\mathcal{E}^0$ as the set of efficient points detected when addressing $\min_{x\in\mathcal{X}\cap\mathbb{Z}^n}(f_1(x), f_2(x))^\top$. The images of points in $\mathcal{E}^0$ are nondominated points of Problem (TOIP) and define the set $\mathcal{M}^0$. The output of `TrIntOpt`, $\mathcal{M} \subseteq \mathbb{R}^3$, is initially set equal to $\mathcal{M}^0$. The starting ε^1 is set equal to $f_3(\hat{x}^0) - \delta$ and we enter in a loop. At every iteration k, the biobjective problem (BOIP^k) is handled, the set $\mathcal{E}^k$ and its image $\mathcal{M}^k$ are computed and $\mathcal{M}$ is enriched by the points in $\mathcal{M}^k$. As it is shown in Proposition 2.5, the points in $\mathcal{E}^k$ are at least weakly efficient. Then, the new value ε^{k+1} is set equal to $f_3(\hat{x}^k) - \delta$, being $\hat{x}^k$ a maximum with respect to f_3 over the set $\mathcal{E}^k$ and we go on until ε^k is less than $f_3(x^*)$, meaning that the whole criterion space has been visited.

In order to prove that `TrIntOpt` detects the complete nondominated set of Problem (TOIP), we need to assume that the objective functions are positive γ-functions, a concept first introduced in [9]. Basically, we need to assume that a positive value exists that underestimates the distance between the image of two integer feasible points of (TOIP), componentwise.

Algorithm 1: Scheme of `TrIntOpt`

```
Input: (TOIP),  $\delta > 0$, k = 1;
Output: Set $\mathcal{Y}_N \subseteq \mathcal{M} \subseteq \mathcal{Y}_{wN}$ of nondominated points of $(TOIP)$;
Compute $x^* \in \operatorname{argmin}_{x \in \mathcal{X} \cap \mathbb{Z}^n} f_3(x)$
Compute $\mathcal{E}^0$ by addressing $\min_{x \in \mathcal{X} \cap \mathbb{Z}^n} (f_1(x), f_2(x))^\top$
Compute $\hat{x}^0 \in \operatorname{argmax}_{x \in \mathcal{E}^0} f_3(x)$
Set $\mathcal{M} = \mathcal{M}^0 = \{f(x) \mid x \in \mathcal{E}^0\}$
Set $\varepsilon^1 = f_3(\hat{x}^0) - \delta$
while $\varepsilon^k \geq f_3(x^*)$ do
    Compute $\mathcal{E}^k$ by addressing $\min_{x \in \mathcal{X}^k \cap \mathbb{Z}^n} (f_1(x), f_2(x))^\top$,
    with $\mathcal{X}^k = \mathcal{X} \cap \{x \in \mathbb{R}^n : f_3(x) \leq \varepsilon^k\}$
    Compute $\hat{x}^k \in \operatorname{argmax}_{x \in \mathcal{E}^k} f_3(x)$
    Set $\mathcal{M}^k = \{f(x) \mid x \in \mathcal{E}^k\}$
    Set $\mathcal{M} = \mathcal{M} \cup \mathcal{M}^k$
    Set $\varepsilon^{k+1} = f_3(\hat{x}^k) - \delta$
    Set $k = k + 1$
end
Return $\mathcal{M}$ Algorithm
```

Definition 2.3 (Positive γ-function)**.** *Let $\gamma > 0$. A function $g : \mathcal{X} \to \mathbb{R}$ is a positive γ-function over $\mathcal{X} \cap \mathbb{Z}^n$ if it holds $|g(x) - g(z)| \geq \gamma$ for all $x, z \in \mathcal{X} \cap \mathbb{Z}^n$ with $g(x) \neq g(z)$.*

Assumption 2.4. *The functions $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, 2, 3$ in Problem* (TOIP) *are positive γ-functions as in Definition 2.3 for some $\gamma > 0$.*

Linear or quadratic functions defined over $\mathbb{Q}^n$ are examples of functions satisfying Assumption 2.4. Table 1, in Section 4.3 in [8], shows some classes of functions for which Assumption 2.4 holds and reports how to compute γ.

Let Assumption 2.4 hold for all the objective functions in (TOIP) with $\gamma > 0$. Let $\delta > 0$ be the input parameter for `TrIntOpt`. In case $\delta > \gamma$ `TrIntOpt` could miss some nondominated points of Problem (TOIP), since the step size δ may be wider than the distance between two nondominated points. On the other hand, if δ is chosen to be less than or equal to γ, we have that `TrIntOpt` is able to detect the complete nondominated set of (TOIP), as shown in the following.

In Proposition 2.5, we first prove that any point detected by `TrIntOpt` is at least a weakly nondominated point. Then, Proposition 2.6 shows that every nondominated point is detected at some iteration of `TrIntOpt`, so that no nondominated point is left undetected, meaning that the set $\mathcal{M}$, output of `TrIntOpt`, is a superset of the nondominated set $\mathcal{Y}_N$.

Proposition 2.5. *Let Assumption 2.1 and Assumption 2.2 hold. Let Assumption 2.4 hold with $\gamma > 0$ and assume that $\delta \leq \gamma$ in Algorithm 1. Let $\tilde{x} \in \mathcal{E}^k$. Then $f(\tilde{x}) \in \mathcal{Y}_{wN}$.*

Proof. Assume by contradiction that $f(\tilde{x}) \notin \mathcal{Y}_{wN}$, namely $x \in \mathcal{X} \cap \mathbb{Z}^n$ exists such that

$$f_i(x) < f_i(\tilde{x}) \quad i = 1, 2, 3. \tag{1}$$

Since $\tilde{x} \in \mathcal{X}^k \cap \mathbb{Z}^n$ we have that $f_3(\tilde{x}) \leq \varepsilon^k$. Therefore, $x \in \mathcal{X}^k \cap \mathbb{Z}^n$ and $f_3(x) \leq \varepsilon^k$, otherwise $f_3(\tilde{x}) < f_3(x)$, getting a contradiction to (1). Since $\tilde{x}$ is an efficient point for Problem (BOIP^k), we have that

$$\nexists\, \hat{x} \in \mathcal{X}^k \cap \mathbb{Z}^n \text{ such that } f_1(\hat{x}) \leq f_1(\tilde{x}), f_2(\hat{x}) \leq f_2(\tilde{x}),$$

with $f_i(x) \neq f_i(\tilde{x}) \quad i = 1, 2$; so that (1) cannot hold. □

Proposition 2.6. *Let Assumption 2.1 and Assumption 2.2 hold. Let Assumption 2.4 hold with $\gamma > 0$ and assume that $\delta \leq \gamma$ in Algorithm 1. Let $y \in \mathcal{Y}_N$. Then $k \in \mathbb{N}$ and $\tilde{x} \in \mathcal{E}^k$ exist such that $f(\tilde{x}) = y$.*

Proof. Let $k \in \mathbb{N}$ be the iteration of Algorithm 1 where it holds

$$\varepsilon^{k+1} < y_3 \leq \varepsilon^k. \tag{2}$$

Note that such value $k \in \mathbb{N}$ exists from the definition of ε^k within Algorithm 1 and since $f_3 : \mathbb{R}^n \to \mathbb{R}$ satisfies Assumption 2.4. Note also that since $y_3 \leq \varepsilon^k$, we have that $x \in \mathcal{X}^k \cap \mathbb{Z}^n$ exists such that $y = f(x)$. Assume by contradiction that $x \notin \mathcal{E}^k$. Two possibilities need to be considered:

i) if we assume that $x \notin \mathcal{E}^k$ as it is not an efficient point for (BOIP^k), we have that $(y_1, y_2) = (f_1(x), f_2(x))$ does not belong to the nondominated set of Problem (BOIP^k). Then, $\hat{x} \in \mathcal{X}^k \cap \mathbb{Z}^n$ exists such that

$$f_1(\hat{x}) \leq y_1,\ f_2(\hat{x}) \leq y_2,$$

with $f_i(\hat{x}) \neq y_i,\ i = 1, 2$. Since $y \in \mathcal{Y}_N$ it must hold

$$y_3 < f_3(\hat{x}) \leq \varepsilon^k.$$

Since Assumption 2.4 holds and $\delta \leq \gamma$, necessarily $y_3 \leq \varepsilon^{k+1}$, that is a contradiction to (2),

ii) if we assume that $x \notin \mathcal{E}^k$ as it has not been detected by the Oracle satisfying Assumption 2.1 used within Algorithm 1, we would have that $(f_1(x), f_2(x))$ belongs to the nondominated set of Problem (BOIP^k) but $\bar{x} \in \mathcal{E}^k$, $\bar{x} \neq x$ exists such that $(f_1(\bar{x}), f_2(\bar{x})) = (f_1(x), f_2(x)) = (y_1, y_2)$. From Proposition 2.5, it holds that $f(\bar{x}) \in \mathcal{Y}_{wN}$. Then, since $y \in \mathcal{Y}_N$, we have that $y_3 = f_3(x) \neq f_3(\bar{x})$ only if $y_3 = f_3(x) < f_3(\bar{x})$. As before, necessarily $y_3 \leq \varepsilon^{k+1}$ and we get a contradiction to (2).

□

Based on the previous lemmata we are able to prove the following.

Theorem 2.7. *Let Assumption 2.1 and Assumption 2.2 hold. Let Assumption 2.4 hold with $\gamma > 0$. Let $\delta \leq \gamma$. Algorithm 1 finds the complete nondominated set $\mathcal{Y}_N$ of* (TOIP) *after having addressed a finite number of single-objective integer programs.*

Proof. By Proposition 2.5 and Proposition 2.6 we have $\mathcal{Y}_N \subseteq \mathcal{M}$. Thanks to Assumption 2.4, choosing $\delta \in (0, \gamma]$ allows the `while` loop to take at most $m = \left\lfloor \left(f_3(\hat{x}^0) - f_3(x^*)\right)/\gamma \right\rfloor$ iterations. Furthermore, taking into account Assumption 2.1, we have that $\mathcal{M}^k$ can be detected after having solved B^k single-objective integer programs.

Then, considering the single-objective integer programs tackled at the beginning of Algorithm 1 for the computation of x^* and $\mathcal{M}^0$, that is $B^0 + 1$, the total number of single objective integer programs addressed by Algorithm 1 is $B^0 + \sum_{k=1}^{m} B^k + 1$. □

2.1 Avoiding the detection of weakly nondominated points

Starting from the ideas presented in [20], we propose to modify Problem (BOIP^k) addressed at Step 7 of Algorithm 1. An additional nonnegative continuous variable $s \in \mathbb{R}$ is introduced to avoid the detection of weakly nondominated solutions along the iterations of `TrIntOpt`. The biobjective nonlinear mixed-integer problem we consider is as follows:

$$\begin{aligned} \min \quad & (f_1(x) - \rho s, f_2(x) - \rho s)^\top \\ \text{s.t.} \quad & f_3(x) + s = \varepsilon^k \\ & x \in \mathcal{X} \cap \mathbb{Z}^n \\ & s \geq 0, \end{aligned} \tag{BOMIP^k}$$

where $\rho > 0$ is an adequately small number (usually between 10^{-3} and 10^{-6}) (see [20]). Despite Problem (BOMIP^k) is a mixed-integer problem, its nondominated set is finite and can be detected by the same oracle used for addressing Problem (BOIP^k). We denote by $\tilde{\mathcal{E}}^k$ the set of efficient points detected by the oracle in Assumption 2.1, when solving Problem (BOMIP^k).

Proposition 2.8. *Let Assumption 2.2 hold. Let Assumption 2.4 hold with $\gamma > 0$. Then, Problem* (BOMIP^k) *has a finite nondominated set. Furthermore, if $x \in \tilde{\mathcal{E}}^k$ then $f(x) \in \mathcal{Y}_N$.*

Proof. Recall that by $\mathcal{X}^k$ we denote the set $\mathcal{X} \cap \{x \in \mathbb{R}^n : f_3(x) \leq \varepsilon^k\}$. From Assumptions 2.2 and 2.4, we have that the set $\{f_3(x) \mid x \in \mathcal{X}^k \cap \mathbb{Z}^n\} \subset \mathbb{R}$ is a finite set. Furthermore, given $\bar{x} \in \mathcal{X}^k \cap \mathbb{Z}^n$, a unique $\bar{s}$ exists such that

$$\begin{aligned} \bar{s} = \quad \text{argmax} \quad & s \\ \text{s.t.} \quad & f_3(\bar{x}) + s = \varepsilon^k \\ & s \geq 0. \end{aligned}$$

Therefore, the nondominated set of Problem (BOMIP^k) is finite. Let $x' \in \mathcal{E}^k$ be an efficient solution of (BOIP^k). From Proposition 2.5 we have that $f(x') \in \mathcal{Y}_{wN}$. Let $(\hat{x}, \hat{s}) \in \tilde{\mathcal{E}}^k$ be an efficient solution of (BOMIP^k) and assume by contradiction that x' dominates $\hat{x}$, with

$$f_1(x') = f_1(\hat{x}) \quad \text{and} \quad f_2(x') = f_2(\hat{x}).$$

Then, $f_3(x') < f_3(\hat{x}) = \varepsilon^k - \hat{s}$, with $\hat{s} \geq 0$. This implies that $s' \geq 0$ exists such that $f_3(x') = \varepsilon^k - s'$ and $s' > \hat{s}$. However, this contradicts the efficiency of $(\hat{x}, \hat{s})$ for Problem (BOMIP^k), as (x', s') is feasible for (BOMIP^k), with

$$f_1(x') - \rho s' < f_1(\hat{x}) - \rho \hat{s} \quad \text{and} \quad f_2(x') - \rho s' < f_2(\hat{x}) - \rho \hat{s}.$$

□

3 Designing sustainable diet plans through tri-objective 0-1 models

Sustainable diets are defined by the Food and Agriculture Organization of the United Nations [2] as "those diets with low environmental impact that contribute to food and nutrition security and to a healthy life for present and future generations". A sustainable diet must therefore be *healthy*, have low *environmental impact* and respect *cultural habits* in order to be acceptable to the population. These issues are often incompatible, for example low-cost diets correspond to high energy density, whereas diets of higher nutrient density and nutritional quality have higher costs. The *healthiness* of food is guaranteed by following the advices of nutritionists and various medical and governmental institutions, which mainly consist of dietary guidelines [23] defining nutrient requirements, recommended nutrient intakes as well as recommended consumption levels of some foods [33].

The *environmental impact* of food production refers to the level of greenhouse gas emissions, the use of land and water resources, pollution, phosphorus depletion and the impact of chemical products such as herbicides and pesticides. *Cultural habits*, i.e. the composition of meals, food preferences and preparation techniques, are strongly influenced by the traditions, beliefs and values shared by a community. They therefore define the structure of each meal and the set of foods and dishes that are considered edible and acceptable [14]. In addition, the attractiveness and variability of meals must also be considered when designing a diet. A meal plan, or menu, consists of the sequence and composition of daily meals over a given period of time. This can be done by selecting dishes from a given set of recipes with a portion size that generally depends on age, weight, gender, and level of physical activity. The design of a menu can therefore be modeled as the assignment of dishes (resources) to given places in a schedule (slots).

The case study considered in this paper is the design of school lunch menus for primary schools in Italy. The set of dishes available to be served was determined by collecting a sample of several Italian primary school menus (children aged 6–11 years) and results in a list of 178 dishes grouped as 66 first-course dishes (in general including pasta or other carbohydrate sources), 75 second-course dishes (in general a source of protein), 35 side dishes (vegetables, potatoes, or salad), fresh fruit, and bread. Tap water is the only beverage allowed for lunch.

Complying with the Italian recommended dietary allowances (RDAs) [30], a fixed portion size for each dish was considered. Recommended ranges for energy and nutrient intakes (fats, proteins, carbohydrates, sugars, fiber, sodium, calcium, iron, and vitamin B12) for each lunch are also obtained from RDAs. To satisfy these requirements, the energy quantity q_i^{en} and nutrient quantities q_i^p of each dish i and nutrient p were calculated from its ingredients using the Italian Food composition and Nutrition database [1]. Other recommendations include limiting or avoiding consumption of some food groups and increasing consumption of others. For example, health authorities recommend eating more plant-based foods, limiting animal products, especially red meat, and avoiding processed meats. To reflect such recommendations, different food groups are defined (white meat, red meat, eggs, fish, dairy, and vegetables) and dishes are assigned to the appropriate food group g. In addition, dishes containing processed meat are not included in the list of available dishes.

The impact of food production on the environment was characterized by some standard consumption-based indicators, such as the carbon, water, and nitrogen footprints. The first is expressed as carbon dioxide equivalent and takes into account all the primary greenhouse gases, i.e., carbon dioxide CO_2, methane CH_4, and nitrous oxide N_2O, emitted during food production. The second takes into account for the freshwater withdrawals required to produce food and the last the pollution of water bodies and ecosystems due to the excess of nitrogen in agricultural production systems. The carbon and water footprints associated to each dish i, denoted by q_i^{cf} and q_i^{wf} respectively, were computed from its ingredients using the database developed in the framework of the EU SU-EATABLE LIFE project [27]. The nitrogen footprint q_i^{nf} associated to each dish i was estimated from its ingredient using the model presented in [18].

We report in Figure 1 the carbon, water and nitrogen footprints of second-course dishes and side dishes containing vegetables. As expected, the second-course dishes containing red meat are the less sustainable ones.

Cultural habits are naturally complied with the choice of the set of recipes from which to select the dishes of the menu, and with the structure of the lunch which must consists of one first-course dish, one second-course dish, one side-dish, fresh fruit and bread. On the other hand, attractiveness and variability of the menu is pursued by fixing the minimum and maximum number of times that each dish, and dishes of the same food group g, can be served in the menu.

The design of a lunch menu over D days can be modelled as the assignment of $D \times 178$ binary variables x_d^i, each one associated to each available dish i and day d, assuming value 1 if the dish i is served in the lunch of the day d, and 0 otherwise.

Nutritional recommendations consists of recommended nutrient intakes and can then be modeled as lower and/or upper bounds on energy and nutrients

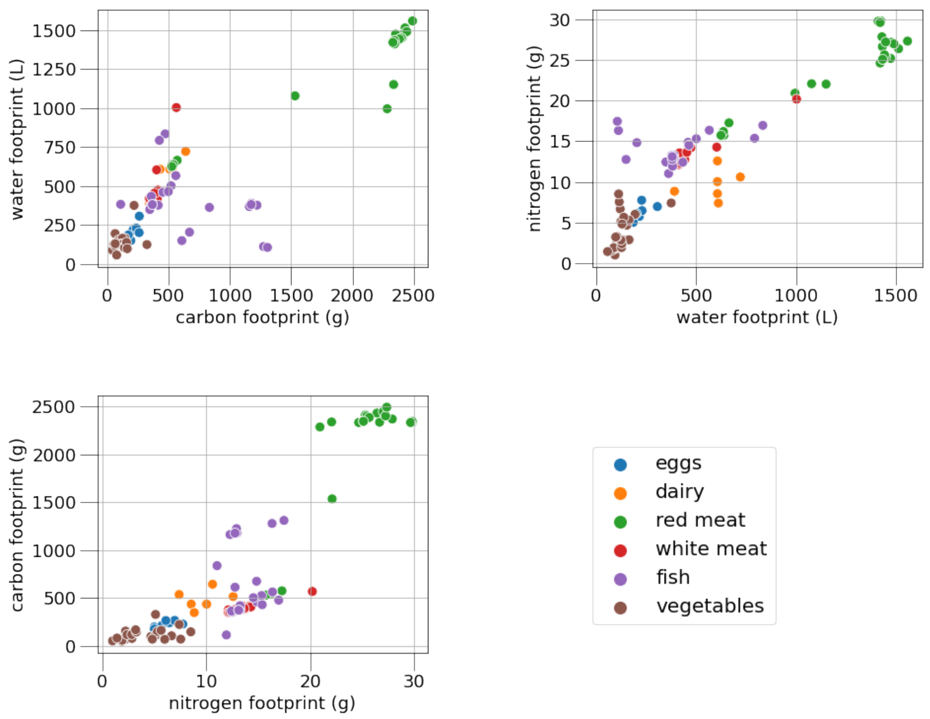

Figure 1: Carbon, water and nitrogen footprints of second-course dishes and side dishes containing vegetables (one fixed-size portion).

contents of each lunch and of the overall menu as follows:

$$L_p^{day} \leq \sum_{i=1}^{178} x_d^i \cdot q_i^p \leq U_p^{day}, \quad \forall d, p$$

$$L_p^{week} \leq \sum_{d=1}^{5} \sum_{i=1}^{178} x_d^i \cdot q_i^p \leq U_p^{week}, \quad \forall p \tag{3}$$

where L_p^{day}, U_p^{day}, L_p^{week}, and U_p^{week} are the lower and upper bounds on the recommended daily and weekly intake of nutrient p. The recommendations limiting or increasing the consumption of some food groups, as well as those related to the attractiveness and variability of the menu, can be modeled as lower and/or upper bounds on the number of dishes of the same food group g served during the week as follows:

$$m_g \leq \sum_{d=1}^{5} \sum_{i \in g} x_d^i \leq M_g, \quad \forall g \tag{4}$$

where m_g, and M_g are the lower and upper bound associated to group g. Moreover, variability is increased by imposing that each dish cannot be served more than once in the menu, that is

$$\sum_{d=1}^{D} x_d^i \leq 1, \quad \forall i \text{ (apart from fruit and bread)}$$

The values of the above defined lower and upper bounds are given in Tables 1 and 2.

p	L_p^{day}	U_p^{day}	L_p^{week}	U_p^{week}
energy ($kcal$)	500	900	3000	4000
fats (g)	10	40	100	150
proteins (g)	15	40	100	175
carbohydrates (g)	70	130	450	550
sugars (g)	-	40	50	150
fiber (g)	-	20	25	75
sodium (mg)	100	700	1500	2500
calcium (mg)	-	-	1000	-
iron (mg)	-	-	25	-
vitamin B12 (μg)	0.35	-	-	-

Table 1: Lunch energy and nutrient constraints for children aged 6–11 years.

g	m_g	M_g
white meat	1	2
red meat	-	1
eggs	1	2
fish	1	2
dairy	1	2
vegetables	4	-

Table 2: Food groups repetition constraints for health, acceptability and variability requirements.

The composition of the lunch of each day d is guaranteed by the following constraints:

$$\sum_{i \in first} x_d^i = 1, \qquad \sum_{i \in second} x_d^i = 1, \qquad \sum_{i \in side} x_d^i = 1, \quad \forall d$$

and $x_i^d = 1$, $\forall d$ and when i corresponds to fruit and bread.

In this paper two different models are considered, with different objective functions and different number of days D. The first model refers to a weekly menu, i.i. $D = 5$, and the objective functions to be minimized are the carbon footprint (expressed as grams of carbon dioxide equivalent emitted), the water footprint (expressed as liters of freshwater withdrawals), and the nitrogen footprint (expressed as grams of nitrogen released) of the menu, that is

$$f_1(x) = \sum_{d=1}^{D} \sum_{i=1}^{178} x_d^i \cdot q_i^{cf}, \quad f_2(x) = \sum_{d=1}^{D} \sum_{i=1}^{178} x_d^i \cdot q_i^{wf}, \quad f_3(x) = \sum_{d=1}^{D} \sum_{i=1}^{178} x_d^i \cdot q_i^{nf}. \quad (5)$$

This model results in a triobjective problem of 890 binary variables and 292 constraints 25 of which are equality constraints. As reported above, the objective functions and the constraints are linear and Assumption 2.4 is satisfied with $\gamma = 0.01$.

The second model refers to a menu for just two days ($D = 2$). In this case the weekly constraints (3) and (4) do not apply and the objective functions to be minimized are the carbon and water footprints of the menu, that is $f_1(x)$ and $f_2(x)$ in (5), and the mean square deviation of the daily energy intake with respect to the RDAs reference value of 700 $kcal/day$, that is

$$f_3(x) = \frac{1}{D} \sum_{d=1}^{2} \left[\left(\sum_{i=1}^{178} x_d^i \cdot q_i^{en} \right) - 700 \right]^2 .$$

The model is then a triobjective binary quadratic one, requiring a much heavier computational burden for our algorithm. This is the reason why the menu runs over just two days, so that only 356 binary variables are needed. Solving this triobjective model ended in identifying 5 nondominated solutions and the related 5 menus, that are reported in the table below. Note that every menu is also including bread and fruit for every day, which are not reported in Table 3.

3.1 Numerical results and discussion

The performance of `TrIntOpt` strongly depends on the performance of the oracle satisfying Assumption 2.1 adopted. In our Python implementation of `TrIntOpt` we solve the biobjective subproblems (BOMIP^k) by the Frontier Partitioner Algorithm (`FPA`) presented in [8]. Assumption 2.1 is satisfied by `FPA` and the number B^k of single-objective integer programs addressed is equal to $|\mathcal{Y}_N^k|+2$, being $|\mathcal{Y}_N^k|$ the cardinality of the nondominated set of the subproblem (BOMIP^k). For how `TrIntOpt` works, it can happen that single-objective integer problems are addressed to detect nondominated points that have already been found. In order to avoid useless computations and save CPU time, we keep a list of the nondominated points detected along the iterations of `TrIntOpt` and use the corresponding efficient points to warmstart the solver of the single-objective integer problems. Within our Python implementation of `TrIntOpt` we use the MIP solver of GUROBI [15]. All experiments have been executed on an Intel(R) Xeon(R) Gold 6252N CPU running at 2.30GHz.

For what concerns the first model, 635 weakly nondominated points were determined. Each point represents a set of menus with the same carbon, water, and nitrogen footprint values, and therefore equivalent with respect to the

		First course	Second course	Side dish
menu 1	Day 1	Soup with rice and cabbage	Frittata	Backed potatoes
	Day 2	Creamy bean pasta	Baked anchovies with breadcrumbs	Green salad
menu 2	Day 1	Creamy potato barley soup	Baked anchovies with breadcrumbs	Cauliflower with anchovy sauce
	Day 2	Creamy bean pasta	Frittata	Stewed early potatoes
menu 3	Day 1	Creamy lentil pasta	Frittata	Stewed early potatoes
	Day 2	Creamy bean pasta	Baked anchovies with breadcrumbs	Spinach with olive oil
menu 4	Day 1	Creamy bean pasta	Frittata	Stewed early potatoes
	Day 2	Pasta with peas	Baked anchovies with breadcrumbs	Cabbage with tomato sauce
menu 5	Day 1	Creamy bean pasta	Baked anchovies with breadcrumbs	Spinach with olive oil
	Day 2	Creamy chickpea pasta	Frittata	Stewed early potatoes

Table 3: Menus obtained solving the second model.

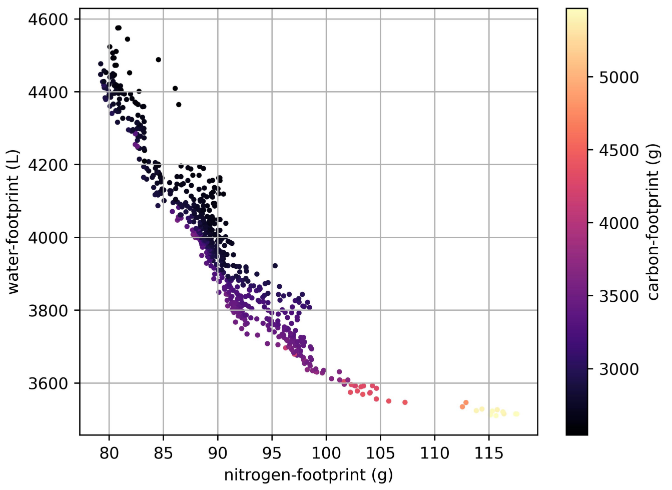

Figure 2: Projection of the weakly nondominated points on the plane nitrogen-water footprints.

environmental impact. For example, the simple permutation of lunches within the days of the week, provides equivalent menus. Moreover, there are dishes sharing the same energy, nutrients, and environmental impact values that can be interchangeably used[1], and this may further increase the number of possible equivalent menus. All the menus associated with the 635 points are nutritionally adequate, healthy and attractive, since they satisfy the constraints.

Projections of weakly nondominated points on the three coordinate planes are shown in Figures 2, 3, and 4.

Figures 2, 3 show that menus with low/high nitrogen or carbon footprint are generally those with higher/lower water footprint. On the contrary, menus with high/low carbon footprint exhibit also high/low nitrogen footprint, see Figure 4. As a consequence, menus with high water footprint have low values for both carbon and nitrogen footprints, and menus with low water footprint have high values for both carbon and nitrogen footprint. This is clearly shown in Figure 4 checking the color mark for the water footprint values. An approximate quadratic relation between the footprint values associated to the (weakly) nondominated points found is obtained by least square fitting with $R^2 = 0.98$, and the corresponding surface is shown in Figure 5. Hence water footprint increases on average quadratically when carbon and nitrogen footprints decrease. This allows, for instance, to estimate the minimum water footprint of a menu fixing

[1]This happens when different recipes have the same ingredients with the same quantity, i.e., they have only a different food preparation such as, for example, Backed potatoes, Boiled potatoes with olive oil, Crispy backed potatoes, and Sauteed potatoes (see Supplementary material).

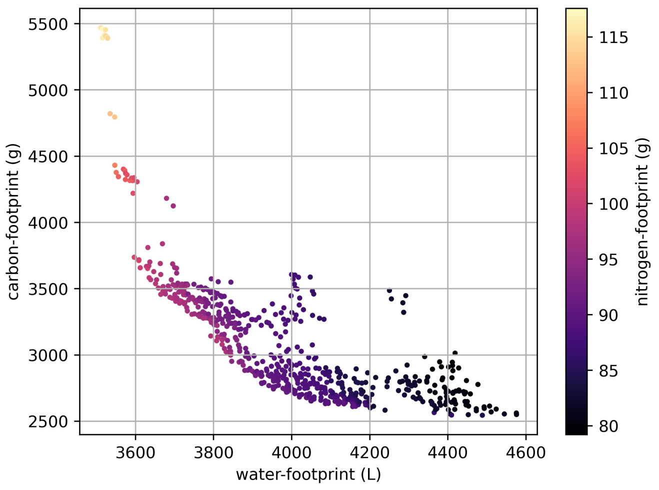

Figure 3: Projection of the weakly nondominated points on the plane water-carbon footprints.

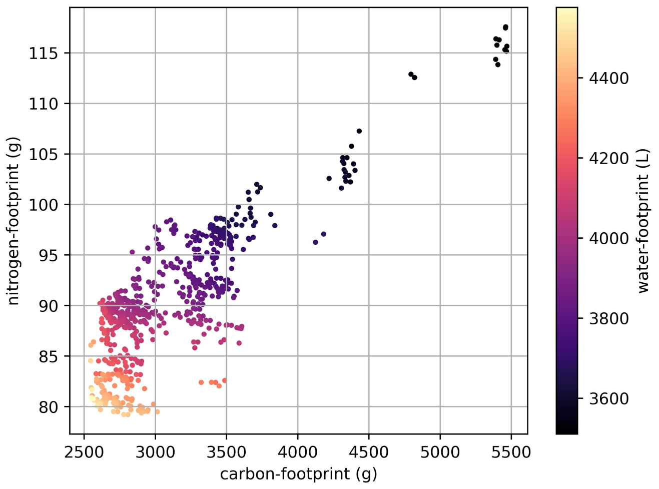

Figure 4: Projection of the weakly nondominated points on the plane carbon-nitrogen footprints.

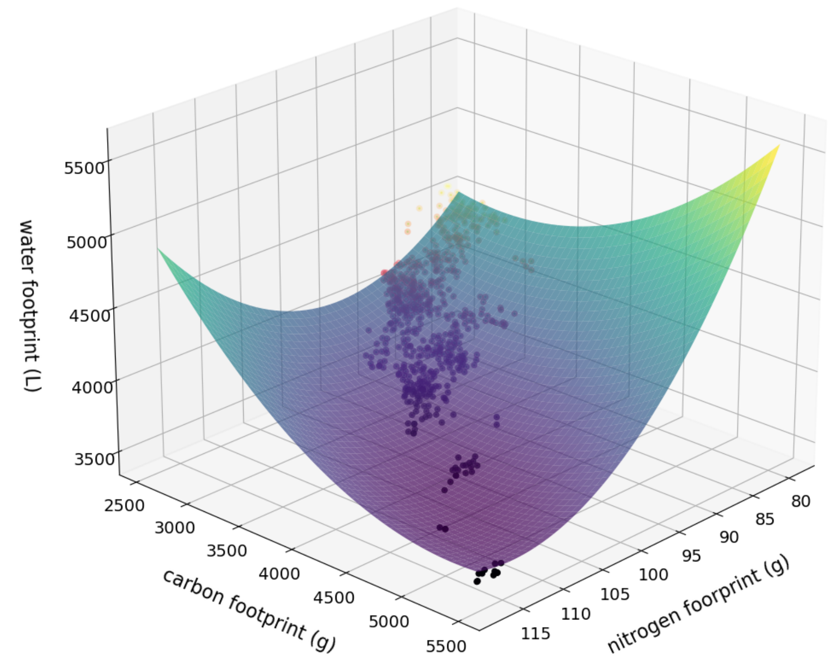

Figure 5: Quadratic approximate relation between weakly nondominated points ($R^2 = 0.98$).

some values for the associated carbon and nitrogen footprint.

4 Conclusions

An exact method for detecting the nondominated set of triobjective nonlinear integer programs has been devised. The method `TrIntOpt` uses the concept of γ-positive function in order to properly combine the ε-constraint method with solvers for biobjective integer programs. A strategy to avoid the detection of weakly nondominated points, seen as a disadvantage of the ε-constraint method, is presented. By applying `TrIntOpt` to two triobjective 0-1 problems modeling the design of nutritionally adequate and healthy diet plans, we were able to collect menus, minimizing standard consumption-based indicators measuring the impact of food production on the environment. Thanks to the theoretical results proven we are guaranteed that each menu detected is a nondominated point of the problem considered and that the whole nondominated set has been recovered.

Appendix: A numerical comparison with LSM [3] and QSM [4] on linear instances

There exists several approaches for multiobjective linear integer (and mixed-integer) programming. We mention the works by Ozlen and coauthors [24, 25, 28] and we refer to [16] for a comprehensive overview on solution approaches for multiobjective mixed integer linear problems. Among the criterion space methods specifically developed for triobjective integer linear programming problems, we find the *L-shape method* (LSM) [3] and the *Quadrant Shrinking Method* (QSM) [4]. In the following, we compare our Python implementation of `TrIntOpt` with both LSM and QSM, that are implemented in `C++` and rely on the IBM CPLEX Optimizer as MIP solver.

The *L-shape method* (`LSM`) [3] is an image space decomposition method. The algorithm holds a priority queue of rectangles to be explored, in the image space of two objective functions. The method looks into the rectangles with the aim of finding all those nondominated points for (TOIP) having their projection in the rectangle itself. In case an already-found nondominated point outside the rectangle is detected the rectangle is shrunk. The search can either determine that the rectangle contains no as yet unknown nondominated point so that the rectangle is discarded, or find a new nondominated point having its projection in the rectangle. In this case, the nondominated point found induces an ”L-shape” which is explored in the same way, namely it can be shrunk, discarded or split into more rectangles that are added to the priority queue.

The *Quadrant Shrinking Method* (`QSM`) [4] partitions the image space by splitting it into “quadrants” which are defined starting from an upper bound point in the image space of two out of three objective functions. A search procedure looks for nondominated points over the current quadrant. Then, if no nondominated point is found the quadrant is shrunk. On the other hand, when a nondominated point is detected, a new search region is defined and the search procedure looks for an as yet unknown nondominated point. Strategies to avoid the solution of redundant single-objective integer problems are implemented.

In Table 4, we report the comparison among $\texttt{TrIntOpt}_{ILP}$, i.e. `TrIntOpt` without the strategy proposed in Section 2.1, `TrIntOpt`, `LSM` and `QSM` on instances of the triobjective assignment problem (AP) available at `https://usf.app.box.com/s/ds0g3ktcjg7vgfsbegzu53ptjgsfq1jg`. Such instances are 0-1 linear and the functions involved have all integer entries, so that Assumption 2.4 for `TrIntOpt` is satisfied with $\gamma = 1$.

For each instance and each method, we report the time needed in seconds and the cardinality of the output set, namely the cardinality of $\mathcal{M}$ for both $\texttt{TrIntOpt}_{ILP}$ and `TrIntOpt` and the cardinality of $\mathcal{Y}_N$ for `LSM` and `QSM`. In fact, despite the adoption of the strategy described in Section 2.1, we noticed that `TrIntOpt` may also detect weakly nondominated points. From our numerical experience, this depends on the choice of the parameter ρ within (BOMIP^k). The results shown are obtained with $\rho = 10^{-6}$, that, in our experiments, leads to the lowest number of additional weakly nondominated points. We can also notice that solving (BOMIP^k), instead of (BOIP^k), comes at a not negligible computational time, as `TrIntOpt` is always slower with respect to `TrIntOpt`$_{ILP}$. This could depend on both the value of ρ and on the introduction of the additional variable s in (BOMIP^k). `QSM` is the fastest method on every instance but AP5 and AP8, where `TrIntOpt`$_{ILP}$ is the fastest. Furthermore, despite `TrIntOpt`$_{ILP}$ detects a higher number of solutions, it is faster than `LSM` on seven instances, suggesting that the warmstarting strategy may pay off.

Instance	TrIntOpt$_{ILP}$		TrIntOpt		LSM		QSM	
	time	$\|\mathcal{M}\|$	time	$\|\mathcal{M}\|$	time	$\|\mathcal{Y}_N\|$	time	$\|\mathcal{Y}_N\|$
AP2	17.18	223	19.20	221	14.46	221	10.75	221
AP3	45.69	500	53.43	488	41.54	483	31.92	483
AP4	260.30	2030	340.38	1962	264.10	1942	226.66	1942
AP5	484.99	3872	617.47	3765	617.29	3750	530.74	3750
AP6	936.99	5380	1097.05	5231	1059.81	5195	874.00	5195
AP7	2169.03	10996	2913.69	10546	2545.78	10498	2231.19	10498
AP8	3289.97	15373	4693.06	14809	4068.96	14733	3552.93	14733
AP9	6971.05	25684	10142.39	24021	7315.51	23941	6626.87	23941
AP10	9437.76	30802	15100.32	29259	10313.50	29192	9337.88	29193

Table 4: Results on triobjective assignment problem instances

Appendix: A numerical comparison with `AdEnA` [13] on quadratic instances

In the following, we report a comparison of `TrIntOpt` with `AdEnA`, the hybrid decision-criterion space method proposed in [13, Algorithm 3] on tri-objective integer nonlinear instances, with convex quadratic objective functions and linear constraints. We used the code of `AdEnA` provided on GitHub [12]. `AdEnA` is an exact method for Multi-Objective Mixed-Integer Nonlinear Problems (MOMINLPs), meaning that it is able to detect an approximation of the nondominated set of a MOMINLP according to a prescribed precision. In particular, `AdEnA` is able to compute an *enclosure* of the nondominated set, i.e. a well-structured set in the image space, as for example a union of boxes, which contains the nondominated set as a subset. The precision of an enclosure as an approximation of the nondominated set is given by its *width* (we refer to [10] for further details on the concept of enclosure). The precision required to `AdEnA` cannot be set to zero, so that the approximation of the nondominated set obtained by `AdEnA` for purely integer instances cannot be compared - in terms of quality - with the exact nondominated set detected by `TrIntOpt`, that is a finite set. We chose to set ϵ, the parameter controlling the width of the enclosure released by `AdEnA`, equal to 0.01. The instances were randomly generated with a number of variables $n = 15$, a number of constraints $m = 10$. The matrices defining the quadratic terms in the objective functions were built using the MATLAB function `sprandsym` and we considered three different density levels $\rho \in \{0.25, 0.50, 0.75\}$. Namely, we generated matrices with approximately $\rho \cdot n^2$ nonzeros entries. For what concerns the linear inequality constraints $Ax \leq b$, we randomly generated matrix $A \in \mathbb{R}^{m \times n}$ and vectors $b \in \mathbb{R}^m$ with $m = 10$ using the MATLAB functions `sprand` and `rand` respectively. For each ρ we produced 5 different instances and we report the CPU time (in seconds) needed by the two algorithms in Table 5. On the instances considered, `TrIntOpt` is not only able to reproduce the finite nondominated set, but it is also almost three times faster than `AdEnA`.

	ρ	time `TrIntOpt`	time `AdEnA`
inst1	0.25	25.69	113.26
inst2	0.25	39.01	102.50
inst3	0.25	55.67	206.12
inst4	0.25	64.98	460.29
inst5	0.25	61.48	439.78
inst6	0.50	180.97	693.57
inst7	0.50	14.22	86.52
inst8	0.50	47.80	195.50
inst9	0.50	11.48	85.94
inst10	0.50	229.15	364.82
inst11	0.75	36.11	164.87
inst12	0.75	41.23	137.59
inst13	0.75	106.97	354.59
inst14	0.75	169.75	404.57
inst15	0.75	268.47	807.92

Table 5: Results on randomly generated triobjective quadratic instances

References

[1] Centro di ricerca alimenti e nutrizione. `https://www.alimentinutrizione.it/tabelle-nutrizionali/ricerca-per-alimento`. Accessed: 19/01/2023.

[2] Sustainable diets and biodiversity: directions and solutions for policy, research and action. In *International Scientific Symposium Biodiversity and Sustainable Diets United Against Hunger*. FAO, 2012.

[3] N. Boland, H. Charkhgard, and M. Savelsbergh. The L-shape search method for triobjective integer programming. *Math. Program. Comput.*, 8(2):217–251, 2016.

[4] N. Boland, H. Charkhgard, and M. Savelsbergh. The quadrant shrinking method: A simple and efficient algorithm for solving tri-objective integer programs. *Eur. J. Oper. Res.*, 260(3):873–885, 2017.

[5] Regina S Burachik, C Yalçın Kaya, and Mohammed Mustafa Rizvi. Algorithms for generating pareto fronts of multi-objective integer and mixed-integer programming problems. *Engineering Optimization*, 54(8):1413–1425, 2022.

[6] Marianna De Santis and Gabriele Eichfelder. A decision space algorithm for multiobjective convex quadratic integer optimization. *Computers & Operations Research*, 134:105396, 2021.

[7] Marianna De Santis, Gabriele Eichfelder, Julia Niebling, and Stefan Rocktäschel. Solving multiobjective mixed integer convex optimization problems. *SIAM Journal on Optimization*, 30(4):3122–3145, 2020.

[8] Marianna De Santis, Gabriele Eichfelder, and Daniele Patria. On the exactness of the ε-constraint method for biobjective nonlinear integer programming. *Operations Research Letters*, 50(3):356–361, 2022.

[9] Marianna De Santis, Giorgio Grani, and Laura Palagi. Branching with hyperplanes in the criterion space: The frontier partitioner algorithm for biobjective integer programming. *European Journal of Operational Research*, 283(1):57–69, 2020.

[10] Gabriele Eichfelder, Peter Kirst, Laura Meng, and Oliver Stein. A general branch-and-bound framework for continuous global multiobjective optimization. *Journal of Global Optimization*, 80(1):195–227, 2021.

[11] Gabriele Eichfelder, Oliver Stein, and Leo Warnow. A deterministic solver for multiobjective mixed-integer convex and nonconvex optimization. Optimization Online, 2022.

[12] Gabriele Eichfelder and Leo Warnow. AdEnA. `https://github.com/LeoWarnow/AdEnA`, 2023. Accessed 2023-09-20.

[13] Gabriele Eichfelder and Leo Warnow. Advancements in the computation of enclosures for multi-objective optimization problems. *European Journal of Operational Research*, 310(1):315–327, 2023.

[14] C. Fjellstrom. Mealtime and meal patterns from a cultural perspective. *Scandinavian Journal of Nutrition*, pages 4161–4, 2004.

[15] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2022.

[16] Pascal Halffmann, Luca E Schäfer, Kerstin Dächert, Kathrin Klamroth, and Stefan Ruzika. Exact algorithms for multiobjective linear optimization problems with integer variables: A state of the art survey. *Journal of Multi-Criteria Decision Analysis*, 2022.

[17] Andre Hugo, Paul Rutter, Stratos Pistikopoulos, Angelo Amorelli, and Giorgio Zoia. Hydrogen infrastructure strategic planning using multi-objective optimization. *International Journal of Hydrogen Energy*, 30(15):1523–1534, 2005.

[18] Allison M. Leach, Ariel N. Majidi, James N. Galloway, and Andrew J. Greene. Toward institutional sustainability: A nitrogen footprint model for a university. *Sustainability*, 6(4):211–219, 2013.

[19] Audrey Legendre, Eric Angel, and Fariza Tahi. Bi-objective integer programming for rna secondary structure prediction with pseudoknots. *BMC bioinformatics*, 19(1):1–15, 2018.

[20] George Mavrotas. Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. *Applied Mathematics and Computation*, 213(2):455–465, 2009.

[21] Pubudu L.W. Jayasekara Merenchige and Margaret Wiecek. A branch and bound algorithm for biobjective mixed integer quadratic programs. Optimization Online, 2022.

[22] Christos A Nicolaou and Nathan Brown. Multi-objective optimization methods in drug design. *Drug Discovery Today: Technologies*, 10(3):e427–e435, 2013.

[23] Institute of Medicine. *Dietary Reference Intakes for Energy, Carbohydrate, Fiber, Fat, Fatty Acids, Cholesterol, Protein, and Amino Acids*. The National Academies Press, Washington, DC, 2005.

[24] Melih Özlen and Meral Azizoğlu. Multi-objective integer programming: A general approach for generating all non-dominated solutions. *European Journal of Operational Research*, 199(1):25–35, 2009.

[25] Melih Özlen, Benjamin A Burton, and Cameron AG MacRae. Multi-objective integer programming: An improved recursive algorithm. *Journal of Optimization Theory and Applications*, 160:470–482, 2014.

[26] Adrián Pascual. Multi-objective forest planning at tree-level combining mixed integer programming and airborne laser scanning. *Forest Ecology and Management*, 483:118714, 2021.

[27] Tashina Petersson, Luca Secondi, Andrea Magnani, Marta Antonelli, Katarzyna Dembska, Riccardo Valentini, Alessandra Varotto, and Simona Castaldi. A multilevel carbon and water footprint dataset of food commodities. *Scientific Data*, 8:127, 2021.

[28] William Pettersson and Melih Özlen. Multiobjective integer programming: Synergistic parallel approaches. *INFORMS Journal on Computing*, 32(2):461–472, 2020.

[29] Gade Pandu Rangaiah and Adrian Bonilla-Petriciolet. *Multi-objective optimization in chemical engineering: developments and applications*. John Wiley & Sons, 2013.

[30] Società Italiana di Nutrizione Umana. *Livelli di Assunzione di Riferimento di Nutrienti ed Energia per la Popolazione Italiana (LARN), 4th ed.*, 2014.

[31] Satya Tamby and Daniel Vanderpooten. Enumeration of the nondominated set of multiobjective discrete optimization problems. *INFORMS Journal on Computing*, 33(1):72–85, 2021.

[32] Aly-Joy Ulusoy, Filippo Pecci, and Ivan Stoianov. Bi-objective design-for-control of water distribution networks with global bounds. *Optimization and Engineering*, 23(1):527–577, 2022.

[33] World Health Organization. *Healthy Diet. Fact sheet N. 394*, 2015.