Operational Research (2022) 22:5231–5267
https://doi.org/10.1007/s12351-022-00702-4

ORIGINAL PAPER

An extensive search algorithm to find feasible healthy menus for humans.

F. Martos-Barrachina[1] · L. Delgado-Antequera[1] · M. Hernández[1] · R. Caballero[1]

Received: 1 July 2021 / Revised: 23 December 2021 / Accepted: 25 February 2022 /
Published online: 23 March 2022

Abstract

Promoting healthy lifestyles is nowadays a public priority among most public entities. The ability to design an array of nutritious and appealing diets is very valuable. Menu Planning still presents a challenge which complexity derives from the problems' many dimensions and the idiosyncrasies of human behavior towards eating. Among the difficulties encountered by researchers when facing the Menu Planning Problem, being able of finding a rich feasible region stands out. We consider it as a system of inequalities to which we try to find solutions. We have developed and implemented a two-phase algorithm -that mainly stems from the Randomized Search and the Genetic- that is capable of rapidly finding an pool of solutions to the system with the aim of properly identifying the feasible region of the underlying problem and proceed to its densification. It consists of a hybrid algorithm inspired on a GRASP metaheuristic and a later recombination. First, it generates initial seeds, identifying best candidates and guiding the search to create solutions to the system, thus attempting to verify every inequality. Afterwards, the recombination of different promising candidates helps in the densification of the feasible region with new solutions. This methodology is an adaptation of other previously used in literature, and that we apply to the MPP. For this, we generated a database of a 227 recipes and 272 ingredients. Applying this methodology to the database, we are able to obtain a pool of feasible (healthy and nutritious) complete menus for a given D number of days.

Keywords Multi-criteria programming · Heuristic integer programming · Algorithms · Menu planning problem · Inequality system

✉ F. Martos-Barrachina
fmeco@uma.es

[1] Programa de Doctorado Economía y Empresa, 2-4 Dpto. Economía Aplicada (Matemáticas), Universidad de Málaga, 29013 Mílaga, España

1 Introduction

Nowadays, the promotion of a healthy lifestyle is a goal among governments all around the world. Policies to ensure the population habits shift towards exercising and an optimal diet are currently -and have been for the last 20 years- the norm at local, regional and national levels in most developed countries (Bröckling et al. 2010). The high prevalence of various non-communicable chronic diseases (NCD) such as diabetes, cardiovascular disease, cancer or hypertension has forced policy-makers to taking action and invest in prevention (Nugent et al. 2018). The reduction of NCDs is a Sustainable Development Goal of the United Nations (UN) for 2030 in order to help prevent premature death. According to a notorious World Health Organization (WHO) report (Waxman 2004), a global strategy for diet and exercise is essential to achieve these goals, and its ramifications must involve regional and local governments and stakeholders, considering specific cultural realities and traditions. These NCDs have four major risk factors: tobacco, alcohol, diet and physical inactivity.

Consequently, diet is understood, in our time, as a crucial part of our lifestyle. The prevalence of obesity -along with undernourishment- is a problem yet to effectively face. The diet followed in many western countries has led to the double burden of malnutrition, defined as 'the coexistence of overnutrition (overweight and obesity) alongside undernutrition (stunting and wasting), at all levels of the population-country, city, community, household, and individual' (Shrimpton and Rokx 2012). So, the paradox of having undernourished and obese people in the same cities at the same time is taking place.

In this context, the need for a diet pattern that is healthy and affordable is as important as it ever was. In fact, the idea of a 'sustainable diet' has been around for at least a decade (Merrigan et al. 2015; Barosh et al. 2014; Burlingame and Dernini 2011). They are defined by the FAO (Lartey 2019) as those with "low environmental impacts which contribute to food and nutrition security and healthy life for present and future generations". In Europe, institutions and policy makers increasingly recognize that European diets need to become more environmentally and economically sustainable, healthier and more nutritious (Rutten et al. 2018).

Using mathematical techniques to design diets in not new in the scientific literature. Usually, George J. Stigler is considered the father of the Diet Problem. In his groundbreaking work (Stigler 1945), he tried to achieve the cheapest possible diet for a year, and was able to obtain, using heuristic mathematical tools, a better -cheaper- solution than his contemporaries. Unfortunately, Stigler's solutions were unappealing and too simplistic (they consisted of just a handful of food items that had to be consumed daily in the same proportions for a whole year) to be adopted by a real person, so (Smith 1959) formulated three different models -sorted by ascending complexity- that included palatability by avoiding excessive repetition, pairing food items that worked well together (i.e. bread and butter) as one and considering actual food choices made by a pool of a few hundred families he observed. It was George B. Dantzig (Dantzig 1949) who created the Simplex

Algorithm (SA) -eventually the standard procedure to solve any Linear Programming (LP) problem- and used it to solve the Stigler's Diet Problem (DP).

Whereas these models presented notorious differences, they were based on the same mathematical ideas. The three of them solved a continuous problem -solutions in grams of ingredients- which objective was to minimize the cost. Since then, many other considerations have been deemed susceptible of becoming the objective function of the problem, such as sustainability, waste, time, caloric intake, or palatability in its many forms. Additionally, in the decade of 1960 the way humans eat, not by devouring raw food items -solution to the DP-, but by structuring them in meals and recipes was first considered. The DP evolved then to a more complex problem usually referred to as the Menu Planning Problem (MPP). It was first developed using mainly recipes and applied to minimize the cost of a nutritious diet for a daily meal in a hospital (Balintfy 1964). In fact, its popularity grew in those years and aspects such as the texture, color and other aesthetic features were included to increase palatability (Gue 1969). By this time, Eugene F. Eckstein, simultaneously developed a model to generate dinners for a student's hostel (Eckstein 1967). It was him who proposed to establish a clear distinction between the DP and the MPP, so from that point on, the problem should be divided between the cattle feeding problem -Diet Problem- and the Menu Planning Problem (MPP) for people -where human idiosyncrasies play a big part- (Eckstein 1970). While the DP is still consistently studied -even for humans-, the MPP, particularly, has become a popular subject among mathematicians, computer scientists and economists.

While the MPP had been most frequently modelled to minimize the cost of a menu -or a series of menus-, given two different sets of constraints, that ensured nutrition and palatability (Leung et al. 1995), the improvements in the computational capacity tipped the scale in favor of innovation. These technical advances allowed new algorithms to work in the development of new Computer Based Menu Planning approaches. In fact, these new computational capacities allowed for new ways to face multi-objective problems, such as the introduction of Artificial Intelligence (Petot et al. 1998) -to assist professionals in assessing menus- with Rule-Based Reasoning and Case-Based Reasoning -and hybrid designs- that improved the level of satisfaction of the solutions obtained (Marling et al. 1999).

Currently, there are a few trends regarding the study of the MPP. (Gerdessen and de Vries 2015) focused in the mathematical qualities of the problem, such as the impact of the selected achievement function. Additionally, the incorporation of new dimensions -where the SHARP (an acronym for Sustainability, Health, Affordability, Reliability and Palatability) approach stands out- (Ivancic et al. 2018; Mertens et al. 2016), appeared, as well as the analysis of the trade-offs between them (Ferrari et al. 2020; Benvenuti et al. 2019; Moraes et al. 2015). Other approaches tackle the individualistic and public perspectives to menu planning, so the problem can be treated as general and its solutions offered to the public (Maillot et al. 2009; Buttriss et al. 2014), or it can be solved considering individual preferences and choices (Toledo et al. 2019; Michel and Burbidge 2019).

Furthermore, the use of a variety of algorithms (or other methods such as Data Envelopment Analysis (Kanellopoulos et al. 2020)) that efficiently find solutions to both the DP and the MPP has recently become a popular choice among researchers

(Pichugina 2020; Funabiki et al. 2011). In the work of (Syahputra et al. 2017) a Genetic Algorithm is used to schedule diets for diabetic patients. (Marrero et al. 2020) compared the use of a Memetic Algorithm (MA) and an Iterated Local Search combined with a Multi-Objective Evolutionary Algorithm based on Decomposition (ILS-MOEA/D) to generate healthy, balanced and inexpensive lunch menu plans for a school cafeteria, while (Hernandez-Ocana et al. 2018) used a Two Swim Modified Bacterial Foraging Optimization Algorithm TSM-BFOA 'to minimize the difference between the number of calories required by an individual and the number of calories provided by the healthy menu found applying TS-MBFOA'. The use of the MPP as the basis for the development of food systems in health care institutions, prisons, schools or the hospitality industry has also been quite common (Lancaster 1992; Sufahani and Ismail 2014; Moreira et al. 2018; Aggarwal et al. 2020).

When facing the Menu Planning Problem -structured by plates and meals- the search for a sufficiently ample feasible set is a challenging task, especially when the problem is scaled, adding more variables and more restrictive constraints (Benvenuti et al. 2019). The models and methods used to find this feasible region and to subsequently solve this problem (MPP) widely vary depending on many factors. However, most of these works have a common core, the use of a regular optimization problem structure. Regardless of the number of objectives, there is a fixed set of constraints that define the feasible region, and the solution will be within this region. An algorithm is then used to find the feasible region and to find inside it a local optimum -and ideally the global optimum- of the objective function(s) among the feasible set. Additionally, from a methodological perspective, as far as we know, there is just another research team working towards (Benvenuti and De Santis 2020) solving the MPP when considered as a whole complete D-day menu where nothing -not a single ingestion- is left out. In any case, most previous work is based on solving the continuous problem (DP) or used -as aforementioned- to schedule meals in hospitality management institutions, such as hospitals (Guala and Marenco 2020), school cafeterias (Segredo et al. 2020; Benvenuti et al. 2016) or nursing homes (Benvenuti et al. 2021; Benvenuti and De Santis 2020).

In contrast, in this work, we take a different angle. Essentially, our approach is different from previous work in the fact that we initially consider a system of inequalities with integer variables defined as conditions -expected qualities of menus- that any proper design of a menu must comply with. Any candidate menu that is a solution to this system must also be a feasible menu in the MPP. To find solutions to this system, we apply an integer optimization problem with no constraints where we pursue to minimize the value of the objective function, defined as a distance function. This distance -always equal or greater than 0- is that calculated from any candidate menu to the inequalities of the system. Due to the lack of public instances, there is no comparison made with alternative results.

Our work goes further with a model that not only presents flexibility regarding many aspects, but also offers D-day menus as the final output. When our algorithm finds valid solutions to the system of inequalities, it is, in fact, finding multiple feasible menus for the MPP. By solving the system, we resolve the underlying feasibility problem -ensuring at least affordability and nutritional adequacy- with a great number of recipes and the patterns of the Mediterranean Diet. This is not an easy task,

given that the MPP -as seen in this work- requires to find a combination of properly structured plates (as integers) that verifies every inequality.

In order to find solutions to our system, we propose a two-step hybrid algorithm inspired in the combination of GRASP -to generate random D-day menus and then, improve them in order to obtain an array of candidate menus that are solutions to our system- with several local searches and the combination of different solutions to take advantage of their good qualities in order to obtain more feasible D-day menus. Due to the lack of test instances, we have created a database of 227 recipes based on 272 ingredients which nutritional values are known. The prices of the ingredients are used according to the *Instituto Nacional de Estadística (INE)*, the Spanish official statistics bureau, when possible. For ingredients which prices are not reckoned by the INE, these were obtained and by checking current prices in Spanish groceries.

Eventually, with just a few tens of seeds -so it can be performed in a timely manner-, our algorithm is able to construct them (about 4 seconds per initial candidate) and improve them (about 4.5 seconds per candidate improved) -averaging more than a 50% of success in this phase. Later, the best candidates to solutions -under a threshold value in the objective function- are combined in two different ways. First, a random interchange of days between different menus -the threshold for a menu to be valid for the following steps is a specific value of the objective function- is performed (0.05 seconds for every menu tried), and second, an exchange of dinners between two menus (0.06 seconds for every valid solution created.). The last step is a shuffle (changing the order of the dinner or lunch plates to ensure daily balance and avoid excessive repetition (0.025 seconds for every modified menu). To sum up, in less than an hour, a few thousand solutions to the system, and therefore, valid menus, are found.

In the following sections of our work, there is an exposition of the database and the structure of the menu used (Sect. 2), the methodology, starting with a thorough description of the model, and a step by step description of the hybrid algorithm employed (Sect. 3), an analysis of the parameters used (Sect. 3.3), an overview of the results obtained (Sect. 4) and the afterward conclusions (Sect. 5).

2 Database and solution structure

Usually, dietary problems (and in particular the DP) have focused on finding an optimal composition, in terms of ingredients' quantity. However, in the Menu Planning Problem (MPP), the solution will specifically indicate all the plates to be consumed day by day. Hence, it is necessary to create a database that links the important elements of this problem: the ingredients, with all their nutritional composition and cost to a list of recipes combining those ingredients. This table will allow us to determine the nutritional composition of each recipe, as well as their composition by groups of foods.

We designed the recipe data set based on a list of I ingredients, characterized by their nutritional profiles (Moreiras et al. 2017) and categorized into different food-groups. In the work we use the words 'plate' and 'recipe' to refer to the same basic

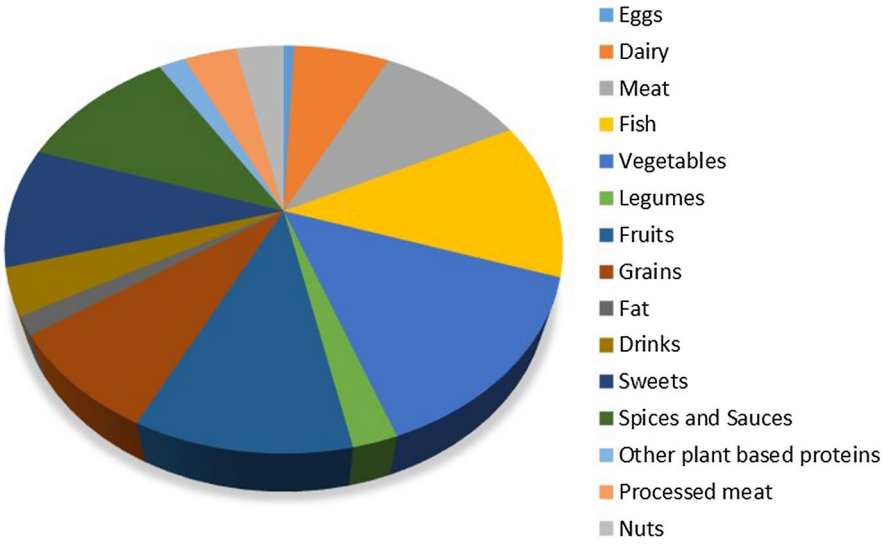

Fig. 1 Food's distribution in groups

Table 1 Definition of standard quantities per type of recipe

Item type	Quantity suggested (g)
Nuts	30
A glass of non-alcoholic drink	200–250
A cup of hot-drink	175–200
Fruits	100
Sweets and dessert	80–100
Sides bread	75–100
Main dishes	200–250

components of our menus. A total of $M = 272$ ingredients, classified into 14 groups, will be the crucial elements to build the recipes. As observed in Fig. 1, the most represented groups are vegetables, fish, spices and sauces and meat, which constitute more than half of the ingredients in the database. On the contrary, the group of eggs, fat and other plant-based proteins are the groups with the least items within the list.

This classification will later be used to define some of the equations in Sect. 3.

As mentioned before, these ingredients form the basis to define the recipes. A total of 227 recipes (N) have been included into our database, using mainly Spanish traditional recipes. For each recipe, the amount of every of its ingredients has been defined. We have established some oriented basic quantities per recipe, inspired by (Moreiras et al. 2017) as shown in the following table:

This scheme, as defined in Table 1, allows us to consider the recipes as units and face the problem from a discrete perspective. Note that, it would be possible

to adjust the size of the recipes or allow some variations of it, but it would likely converge into a continuous problem. Anyway, as the composition of the recipes is given by the percentage of each ingredient required, their nutritional profiles are easily determined by multiplying the composition of the recipes by their standard quantity, and the result by the nutritional profile of the ingredients. This procedure facilitates the calculation of the nutritional composition for each recipe. For instance,the amount of protein, Pr_n (in Kcal) in a specific plate n, is calculated by using the amount of each ingredient in its recipe ($q_{m,n}$, $m = 1, 2, \cdots M$, in grams) and the amount of protein Pr_m (in Kcal) present in the m^{th} ingredient.

$$Pr_n = Q_n \cdot \sum_{m=1}^{I} q_{m,n} \cdot Pr_m \ \ \forall n = 1, 2, \cdots N \tag{1}$$

Once the elements of the menu have been set, the next step is to design a well-defined structure for a D-day menu plan. In particular, our work contemplates the design of a Spanish dietary plan -although we are working towards implementing flexibility-. Obviously, it is an unattainable task to establish a common structure for the Spanish population. In previous related works a menu consisting of breakfast, lunch and dinner (Toledo et al. 2019) and the inclusion of one or two snacks (Benvenuti et al. 2019; Hernandez-Ocana et al. 2018) has been used. In our work, we are only including one daily snack, but it can be easily modified to omit it or to include a second one. However, some patterns are nationally accepted and have been considered in this work. This includes a breakfast, a lunch -most copious meal in Spain-, a dinner and a snack. Nevertheless, there are a few not-explicit but well established "rules" that define the common pattern. In this case, in order to create a menu plan for a D-day period, we have taken into consideration the following daily structure, where k indicates the consumption pattern:

It can be observed in Table 2 that we have numbered -k- the intakes that happens throughout the day to ease the formulation. This gives us a hard structure that cannot be dismissed so it has to be respected by any menu candidate to be a solution.

Table 2 Definition of menu daily-structure

Intake	Consumption pattern	k
Breakfast	Hot beverage	1
	Juice or piece of fruit	2
	Breakfast dish	3
Lunch	Small bun of bread	4
	Cold drink	5
	Starter	6
	Main dish	7
	Dessert	8
Supper	Cold drink	9
	Dinner	10
	Dessert	11
Extra	Snacks	12

Hence, in order to formulate a solid structure for any solution, a daily menu will be given by an array of length 12, formed by the corresponding items consumed in a day. To facilitate the procedure, we have labelled every recipe to indicate if it is suitable for breakfast, main dish, dinner or snack. Also, there is a subdivision between hot beverages and cold drinks.

Figure 2 gives an idea of the variety of recipes considered in this study. It is customary in the Spanish pattern to give more importance to lunch and dinner than to breakfast. As a result, there is a larger number of recipes available for those meals in the database and the variety for breakfast is shorter. Around 50% of the recipes are an option for lunch and 22% for dinner, whereas just 3% are breakfast options. Besides, the set of fruits is enough to include different alternatives. Finally, we need to mention that, in the development of the database, the authors have decided not to include several recipes because their composition were not compatible with some of the basic nutritional constraints. Therefore, most of these rejected recipes were very unlikely to be in a feasible menu so their presence in the database would only distract the search procedure and slow the algorithm. This is the case of cakes, fast food items and others.

To sum up, a solution for a D-day menu plan (or a *menu* from now on), will be a matrix where the rows represent the meals and the column, the suggested day. Additionally, following the Spanish custom of eating with bread, a small bun -whole or regular- will be consumed with lunch. For instance, the following table (Table 3) is a part -3 days- of a randomly generated plan, given the structure defined at Table 2.

As mentioned, any candidate menu will suggest eating a bun of bread with lunch. This realistic menu, indicates that the composition of the supper for day 2 to be a can of diet coke, an omelet and a yoghurt for dessert as well as some nuts as snacks; a Spanish soup as a starter in day 3 with a glass of water or a steak with a side of salad in day 3.

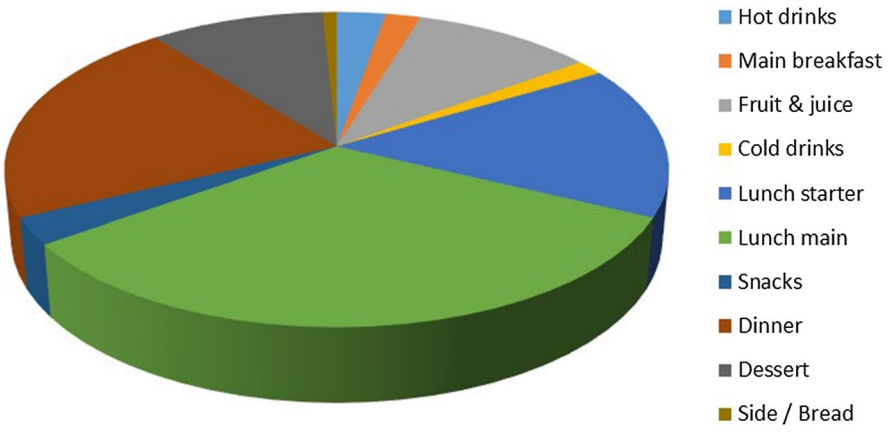

Fig. 2 Recipes classification into consumption patterns

Table 3 Example of candidate menu structure

Consumption pattern	Day 1	Day 2	Day 3
Juice or piece of fruit	Portion of Strawberries	Apple	Orange juice
Hot Beverage	Coffee w/ semi skimmed milk	Cocoa	Black coffee
Breakfast dish	Breakfast cereals w/milk	Ham & cheese sandwich	Toast with butter
Small bun of bread	Regular bread	Whole wheat bread	Whole wheat bread
Cold drink	Glass of wine	Water	Water
Starter	Tomato salad	Cold almond soup	Spanish soup
Main dish	Cod a Bras	Spanish 'Tortilla'	Steak with side of salad
Dessert	Custard	Yoghurt w/ raspberries	Banana
Cold drink	Water	Diet coke	20 cl of beer
Dinner	Omelette	Chicken breast w/ rice	Grilled tuna w/salad
Dessert	Portion of apricot	Yoghurt	Portion of peach
Snacks	Almond	Nuts	Mix of nuts

3 Methodology

The aim of this work is to design an algorithm able to assess complete menus satisfying the nutritional requirements for an individual and which general frames respect the Mediterranean standards. A priori, these requirements are given as an inequality system, with several binary variables that contemplate not only the recipe but also the exact day and meal of the intake.

Then, in order to find a solution, the inequality system is transformed into an optimization problem with no constraints. Thus, we define a single-objective menu plan model that contemplates the boundaries established by the system. This function aims to minimize the distance of the randomly generated menus -using our database and respecting the structure- to the limitations defined by the inequalities. In this case, despite different distances could be applied, the one that guarantees more consistent and balance solutions is the weighted Tchebycheff procedure (Steuer and Choo 1983). Note that a solution is a complete menu plan for D-days whose objective function is null, i.e., it satisfies each inequality of the system of requirements. It is important to notice that several solutions may satisfy the set of constraints, despite their different composition. Besides, an additional array counts the frequency of appearance of each of the plates from the dataset, so we are able to check if a solution is different from another.

3.1 The model: variables, constraints and objective function

The goal is to design complete menus that offer a whole plan (with every meal) for D days that specify everything that should be eaten (see Table 3) day by day. There is a vast number of possible combinations, so it is interesting to generate a sufficient number of candidates -as many as possible- that satisfy the given set of nutritional and consumption requirements, which will form a set of alternative menu plans.

The model used in this work coincides with the one recently presented by (Benvenuti et al. 2016, 2019), when designing menus for school lunches first and for nursing homes later. In (Benvenuti et al. 2016) a model for school lunches is described, using a set of 106 Italian recipes -prepared with 71 ingredients- already defined and in use by the municipality of Rome. The optimal combination of dishes would offer adequate nutrition while minimizing the greenhouse gas emitted and the water consumed.

The model defined in (Benvenuti et al. 2019) is more ambitious, and creates 2-week menu plans for a nursing home with 140 recipes. They use the ϵ-constraint method to find the pareto front having cost and emissions as the conflicting goals. We go to a bigger scale with more variables and constraints, and therefore have to use a heuristic algorithm -because of the scale- to find solutions. However, Italian and Spanish palates and meal structures are similar, as both are Mediterranean, so our solutions have, at least, these points in common.

In fact, in the latter, the difficulty of finding a feasible set when scaling the problem is referred as 'scalability of the model significantly impacts on the number of variables and constraints, thus delivering optimization problems with increasing size. This may produce a very long computation time to solve the optimization problems but a more serious limitation that may occur is the downsize of the set of feasible solutions when more constraints are considered'.

In direct answer to this, the main contribution of our work is the scaling of the problem and the use of a hybrid algorithm to explore the feasible set and enrich it. In our work we include macro and micro nutritional constraints and additional constraints to guarantee MD standard intakes of fish, nuts, legumes, vegetables and fruits as well as including drinks in a set of 227 recipes and 272 ingredients.

To design the mathematical model for the menu planning problem, one solution will provide information not just about the quantity of the recipes to be consumed, but also the specific day and meal. To face this matter, some notation aspects are detailed in the following lines:

- N determines the number of recipes available.
- $n = 1, 2, \cdots N$ corresponds to the list of plates, as read from the data set. So that, the size of N is equal to the number of available recipes.
- $m = 1, 2, \ldots M$ refers to the number of ingredients that conform the available recipes.
- $d = 1, 2, \cdots D$ represents the day in a time horizon of D days.
- $i = 1, 2, \cdots I$ refer to a specific constraint -out of the I active ones-.
- $k = 1, 2, \cdots K$, where K corresponds to the intakes of the structure, as mentioned in Table 2. It allows to identify if a concrete item is taken for lunch or dinner, for example.
- $X_n^{d,k} \in \{0, 1\}$ takes the value 1 if the n-th plate is consumed the d-th day for the k-th meal time. These are the decision variables and will help to complete the structure of the menu plan for D-days.
- F_n represents how many times is the n-th plate suggested within the same solution. It seems obvious that it can be calculated as:

$$F_n = \sum_{d=1}^{D} \sum_{k=1}^{K} X_n^{d,k} \tag{2}$$

- C_n is a constant array that contains the recommended quantity (in grams) of the n-th plate to be consumed, according to the portions established in Table 1.

Taking all the above into consideration, a solution is a combination of recipes daily organized into a menu structure as defined by Table 3. This composition is given by the binary variables $X_n^{d,k}$, whose interpretation provides a suggestion of a menu -expressed in given recipes- for a fixed period. Besides, as observed, these decision variables might be aggregated into the frequency that each plate is consumed, F_n. However, in order to simplify the notation, an auxiliary variable (Q_n) is defined to represent the quantity (measured in grams) of the nth plate consumed along the menu. Its calculation is easy:

$$Q_n = F_n \cdot C_n \ \forall n = 1, 2, \ldots N \tag{3}$$

Now, we construct the system of inequalities which is based in two pillars: the essential nutritional requirements and the Mediterranean diet standards. On the one hand, nutritional requirements are defined according to a particular given profile. As mentioned in Sect. 1, there are many situations that determine the boundaries for the macro-nutrients and micro-nutrients that have previously been studied. Despite it can be adapted to any profile, we consider as a reference for any shown solution, a healthy moderately active woman in her 30's, so that the set of nutritional requirements are (by day) the inequalities in the following general structure according to the relation to the right-hand side (Moreiras et al. 2017):

$$\begin{cases} R_U = \{A \cdot Q \leq b_U\} \\ R_L = \{A \cdot Q \geq b_L\} \end{cases} \tag{4}$$

Therefore, most of the specific values for the array of constraints are showed in the tables that follow. We consider the coefficient matrix A that summarize the inequalities below expressed as $A \cdot Q \leq b_U$ or $A \cdot Q \geq b_L$, where the index determines if the inequality is defined with the constant term belongs to R_U or R_L. Note that for a double inequality, such as Eq. 5, it must be split into two different equations:

$$1840 \leq \sum_{n=1}^{N} E_n \cdot Q_n \leq 2300 \Rightarrow \begin{cases} \sum_{n=1}^{N} E_n \cdot Q_n \geq 1840 \\ \sum_{n=1}^{N} E_n \cdot Q_n \leq 2300 \end{cases} \tag{5}$$

We are solving an instance of the MPP for D=15 days where almost every constraint is applied for the whole period of 15 days. However, the values of the boundaries of the constraints are shown in its daily amount. These daily boundaries intend to

illustrate the scaling possibilities of the problem to $D = D_0 \; \forall D_0 = 1, 2, \ldots \mathbb{N}$. The only boundary that is used daily is in the caloric balance constraint in Eq. 14.

It is worth to note that we have a number I of inequalities with similar structures that have to be satisfied. Following these steps, we can divide the system into two groups:

$$\sum_{n=1}^{N} A_{i,n} \cdot Q_n \leq b_{U,i} \tag{6}$$

where $b_{U,i}$ denotes the upper bounds $\forall i \in R_U$, i.e. for inequalities described as $A \cdot Q \leq b_U$.

$$\sum_{n=1}^{N} A_{i,n} \cdot Q_n \geq b_{L,i} \tag{7}$$

where $b_{L,i}$ denotes the constant term $\forall i \in R_L$, i.e. for inequalities described as $A \cdot Q \geq b_L$. In the previous example, the corresponding $b_{U,1} = 2300$ and $b_{L,1} = 1840$.

In particular, all the boundaries previously mentioned are summarized in Table 4. Most micronutrients have just a minimum or a maximum amount and not both. A limit that is not present in the table represents a lack of a maximum or a minimum required amount. The nutrients are expressed in daily amounts, but the computed model uses only the global amount -multiplying the daily amount for the number of days of the instance- to find solutions. In the presented instance,

Table 4 Nutritional Daily Requirements

Nutrient	Lower bound (b_L)	Upper bound (b_U)
Energy (Kcal)	1 840	2 300
Fiber (g)	25	–
Cholesterol (mg)	–	300
Calcium (mg)	750	–
Iron (mg)	14	–
Magnesium (mg)	330	–
Sodium (mg)	–	2 000
Potassium (mg)	2 000	–
Phosphorus (mg)	700	–
Niacin (mg)	15	–
Folate (μg)	400	–
Vitamin B12 (μg)	2	–
Vitamin C (mg)	80	–
Vitamin A (μg)	800	–
Vitamin D (μg)	5	–
Vitamin E (mg)	12	–
Water (g)	2 000	–
Cost (Euro)	–	10.50

with 15 days, the fiber consumed has to be greater than 375 grams. Sodium, for instance, is constrained to less than 30.000 milligrams for the 15-day period. All the micronutrients are treated similarly.

Note that the coefficients of these inequalities depend on the ingredients and can be calculated, as defined in Sect. 2, by multiplying the amount of each ingredient specified at each recipe, by their corresponding nutritional composition. Besides, these boundaries are given on a daily basis, so that for a D-day horizon menu plan, these lower and upper bounds should be multiplied by the number of days, D, as we have done in our work. This means that the constraints do not have to be satisfied daily, but for the duration of the whole planning period. Additionally, we consider a set of restrictions that limits the energy (Kcal) that can be obtained from any macro-nutrient. The macro-nutrients amounts are hereby considered also in Kcal. However they are expressed as a proportional amount of energy when computed in the constraints, so they do not depend on the number of days.

As can be seen in Table 5, there is a threshold for every macronutrient. Proteins have to represent between 10 and 20 percent of all energy consumed, while fats have to be between 25 and 40 percent. Carbohydrates are the core energetic source, with 40–60% of total intake, but limiting sugar to a maximum of 15%.

There are two other conditions related to the quality of the fat consumed. The fat can be saturated (SF), monounsaturated (MUF) or polyunsaturated (PUF). The amounts of Fats are expressed in grams and, according to (Moreiras et al. 2017), the ratio between them should be:

- SF & PUF

$$\frac{\sum_{n=1}^{N} PUF_n \cdot Q_n}{\sum_{n=1}^{N} SF_n \cdot Q_n} \geq 0.5 \tag{8}$$

  where PUF_n denotes the percentage of polyunsaturated fat given by each gram of the n-th plate and, similarly, SF_n the saturated fat.
- SF, MUF & PUF

$$\frac{(\sum_{n=1}^{N} MUF_n \cdot Q_n + \sum_{n=1}^{N} PUF_n \cdot Q_n)}{\sum_{n=1}^{N} SF_n \cdot Q_n} \geq 2 \tag{9}$$

Table 5 Macronutrients to total energy

Macronutrient (% of Total Kcal)	Lower Bound	Upper Bound
Protein	10	20
Carbohydrates	40	60
Sugar	–	15
Fat	25	40

where MUF_n denotes the percentage of monounsaturated fat given by each gram of the n-th plate

As previously mentioned, in Sect. 2, we propose a model inspired by the Spanish customs and, due to the geographical situation and the culture of this country, our model considers an additional set of inequalities related to the Mediterranean Diet. Besides, the adherence to this diet is related to great benefits in human health (Trichopoulou et al. 2003; Willett et al. 1995; Bach-Faig et al. 2010), among other qualities, such as economic or environmental sustainability (Germani et al. 2014). In this context, the restrictions considered in this work are related to the total quantity (in a day) of foods classified into 14 groups in Sect. 2, as follows (Hernández et al. 2019):

Table 6 is showing us the allowed minimum or maximum amounts of this specific food groups. Although these amounts are expressed daily, the model considers only the whole period of time and therefore, the amounts adapted to that period. In this case, for instance, processed meat has a very restrictive maximum amount of 20 grams a day. However, in our presented instance, with a horizon of 15 days, this constraint is modified -multiplying said amount for the number of days- and the computed constraint limits processed meat to less than 300 grams for the 15-day schedule. Red Meat, Fish, Extra Virgin Olive Oil and the rest of groups presented in this table are treated similarly.

- Red Meat & White Meat

$$\sum_{n=1}^{N} RM_n \cdot Q_n \leq \sum_{n=1}^{N} WM_n \cdot Q_n \tag{10}$$

where WM_n and RM_n denote the amount of white meat and red meat contained in the n-th plate.

- Meat & Fish

$$\sum_{n=1}^{N} Meat_n \cdot Q_n \leq \sum_{n=1}^{N} Fish_n \cdot Q_n \tag{11}$$

Table 6 Constraints of Food Items

Food Item (or Group)	Lower Bound	Upper Bound
Red Meat (g)	–	50
Processed Meat (g)	–	20
Fish (g)	40	–
Vegetables (g)	300	–
Fruit (g)	250	–
EVOO (g)	25	–
Butter (g)	–	10
Legumes (g)	6.5	–
Nuts (g)	22	–
Sweets (g)	–	15

where $Meat_n$ and $Fish_n$ denote the amount of meat and fish in the n-th plate. Note that this meat variable reflects any kind of meat, whether it is white, dark or processed meat.

Additionally, for each recipe, we refrain any of them from appearing more than T times (Eq. 12) in the whole plan with the exception of drinks, breakfast items and bread buns. These exceptions are considered because, culturally, the repetition of these items does not interfere with palatability. These conditions can be formulated as follows:

$$\sum_{d}\sum_{k} X_n^{d,k} \leq T \tag{12}$$

All these conditions constitute the system of inequalities. Any point inside this set represents a menu structured by daily intakes for a D-day period, that satisfies all the requirements presented in the inequalities. Then, our goal is to find as many of those points -which verify all the conditions- as possible. The more the better.

Now, our approach constitutes a method to solve a mathematical programming problem with binary variables, which optimizes a function that represents the distances between both members of the inequalities and has no constraints. If this function reaches the null value for a particular point, it will mean that the point verifies the system of inequalities.

As mentioned before, to evaluate the distance we consider weighted Tchebycheff metric (Steuer and Choo 1983). It considers a correction factor, defined as a very small constant ρ (with a value between .01 and .05), to avoid generating weakly efficient solutions. Its definition is based on the Tchebycheff distance, which takes the maximum among all the inequalities violations for a menu; and applies the correction factor to the Manhattan distance, which is defined as the sum of all the normalized inequalities violations for a menu. Hence, we define our objective function as follows:

$$\begin{aligned}\text{Min}\Bigg\{ &\max_{n=1\ldots N}\left\{\sum_{i\in R_U}\frac{1}{b_{U,i}}\left(A_{i,n}\cdot Q_n - b_{U,i}\right), \sum_{i\in R_L}\frac{1}{b_{L,i}}\left(b_{L,i} - A_{i,n}\cdot Q_n\right)\right\} \\ &+\rho\cdot\left(\sum_{i\in R_U}\frac{1}{b_{U,i}}\left(A_{i,n}\cdot Q_n - b_{U,i}\right) + \sum_{i\in R_L}\frac{1}{b_{L,i}}\left(b_{L,i} - A_{i,n}\cdot Q_n\right)\right)\Bigg\}\end{aligned} \tag{13}$$

Hence, by definition, the objective function pushes the candidate menus towards the solution set, with the aim to reach the non-violation scenario, which corresponds to a null value of this objective function.

Additionally, to avoid daily imbalance, a solution must comply with another two constraints. First, regarding energy, the daily intake must be inside a ±25% threshold of the energy constraint, so additionally to complying throughout the menu with the global energy constraint, a daily balance has to be maintained. To enhance diversity in the intakes, another requirement does not allow to eat fish or meat -repeat protein- during both lunch and supper on the same day (Eq. 15). This means that to enhance

palatability, the repetition of fish and the repetition of meat cannot take place in any day. These two final conditions are encountered in the final phase of the algorithm.

Daily Energy (Kcal)

$$1380 \leq \sum_{n=1}^{N} \sum_{k} E_n \cdot X_n^{d,k} \leq 2944 \quad \forall d = 1, 2, \ldots D \tag{14}$$

where E_n indicates the energy provided for each gram of the n-th plate consumed.

$$\begin{cases} \left(\sum_n \sum_{k \in Lunch} \ Meat_n \cdot X_n^{d,k}\right) \cdot \left(\sum_n \sum_{k \in Dinner} \ Meat_n \cdot X_n^{d,k}\right) = 0 \ \forall d = 1, 2, \ldots D \\ \left(\sum_n \sum_{k \in Lunch} \ Fish_n \cdot X_n^{d,k}\right) \cdot \left(\sum_n \sum_{k \in Dinner} \ Fish_n \cdot X_n^{d,k}\right) = 0 \ \forall d = 1, 2, \ldots D \end{cases} \tag{15}$$

3.2 The algorithm

The MPP, as it is conceived in this work, is -a priori- a collection of inequalities that represents the conditions that a menu must verify. The resolution of this problem is tackled as a mathematical programming problem, with the objective of finding as many points as possible that satisfy the given conditions (Sect. 3.1). Normally, the MPP formulations are modelled as combinatorial problems which are proved to be NP-complete (Gazan et al. 2018), and therefore, cannot be solved by exact methods. So, we propose a methodological approach in two-steps, combining and adjusting different heuristics as stated in the following pseudo-code:

Algorithm 1 General procedure: 2-stage algorithm.

```
1: procedure ALGORITHM(constantRandom,LS).
2:     Stage 1: Generate a set of solutions: S = GRASP(constantRandom).
3:     Stage 2: Densify S applying iterated local Searches and combinations: S' = LSP(S).
4: end procedure
```

It begins by applying a Greedy Randomized Adaptive Search Procedure (GRASP) (Feo and Resende 1995) and find a small set of candidate solutions. Later, these solutions are combined within a Local Search Procedure (LSP) structure that, inspired in the Crossover Operators of Genetic Algorithms, contributes to find new solutions in a short time. Both heuristics has been successfully applied to many other combinatorial real problems. However, to the best of our knowledge, none of them has been used with the aim presented in this work. By definition, GRASP is a multi-start procedure where each iteration requires two steps: the *construction* and the *improvement*. The aim of the first stage is to, following an iterative process, construct a complete solution beginning at a partial solution or *seed solution*. In addition to this, in our study, the solution must fall into an established structure. Then, our method begins with the random generation of a set of initial seeds that conform a part of the daily menu. The greedy

function considered in this procedure is the one defined by Eq. 13. The application of GRASP implies considering a restricted candidate list, *RCL*, which might be defined either as a percentage of the best known value of the objective function (Resende and Ribeiro 2003) or with a prefixed length (Prais and Ribeiro 2000, 1999). In this work, the last approach is taken into consideration, so that *RCL* contains a constant number (*constantRandom*) of alternatives ranked from better to worse based on a hypothetical objective value if the movement took place. In other words: when inserting a new recipe in the menu, the new compositions are evaluated and sorted according to their initial f-value. So, based on the initial seeds, the menus are pseudo-randomly completed by suitable good candidates, from the pre-determined *RCL*, that move the menu towards the region defined by the system (Eq. 6 and 7). In fact, once a menu becomes a solution, thus the objective function (Eq. 13) becomes null, the algorithm stops. Then, the procedure continues with another iteration, trying to find another solution considering a different seed. It is important to recall that, because of the definition of the problem, the goal is to find those menus whose objective function value is 0.

Algorithm 2 Step 1. GRASP.

```
1:  procedure GRASP(constantRandom)
2:      S = ∅
3:      for it = 1 : numIter do
4:          randomSeed = GenerateSeed();
5:          s = ConstructSolution(randomSeed);
6:          s' = Improvement(s);
7:          if f(s') = 0 then
8:              S = S ∪ {s'};
9:          else
10:             if f(s') ≤ f(s) then
11:                 s = s';
12:                 f_best = f(s');
13:                 S = S ∪ {s'}
14:             end if
15:         end if
16:     end for
17:     return S
```

The algorithm begins generating a *random seed*, where a random lunch is created for each day of the planned menu, so the corresponding decision variables $X_n^{d,k}$ for $k = 4, 5, 6, 7$ are randomly assigned (check Table 3 for the structure information). It is a cold drink, one first and main dish and one dessert. Then, the method evaluates the value of the current, and non-complete, menu by Eq. 13. Then, *constructSolution* completes the given *random seed* by introducing recipes in the empty spaces of the structure for each day. As mentioned, the introduction criteria is based on a greedy operator, so that we test what would be its value if any candidate were included in the solution. When all the candidates are evaluated, the *constantRandom* best candidates define the *RCL* and the algorithm randomly selects one. Note that, during the construction of the first complete solution, the algorithm contemplates the following order to fill a gap in the menu:

1. Drinks: hot beverage for breakfast ($k = 1$) and cold drink and for dinner ($k = 9$).
2. Breakfast solid ($k = 3$).
3. Juice or piece of fruit for breakfast ($k = 2$).
4. Supper main dish ($k = 10$).
5. Supper dessert ($k = 11$).
6. Extra snacks ($k = 12$).
7. Extra bun of bread ($k = 4$).

Every time a recipe is introduced into the solution, the current value of the objective function is updated. In addition to this, as stated in Sect. 2, in order to facilitate the calculation, each recipe has been classified, so that for a given k, the candidates to be considered must be allowed to be part of the menu in the kth position of the daily structure.

Now, the second step of GRASP tries to improve the generated solution, by a short local search process. It consists of evaluating the unsatisfied inequalities and semi-randomly, again using a *RCL*, switching one recipe among those menu items that contribute the most to the violation, for another that brings the current menu closer to the satisfaction of all the inequalities. In other words, the improvement procedure consists of testing if the objective value would be better if introducing another plate in the menu, when removing one of the current items that largely contribute to the violation on any of the conditions. Two different approaches are considered in order to define a criterion to interchange recipes in a given menu. These two approaches are sequentially used to improve every menu, in the following order:

- The recipe with a maximum violation is selected to be removed from the menu. Now, in order to introduce another recipe, we consider a list of candidates (*RCL*) with the best α - elements. These are the recipes that would improve the objective function value the most. Then, one of these elements is randomly chosen to be inserted into the menu, replacing the one removed. And so, both of them should belong to the same category of plates, i.e., both available for lunch, for example. This is used first to broadly improve the random seeds.
- Alternatively, another approach is to find the inequality which is the furthest to be satisfied for a particular menu and, again, generate a list of candidates of plates, *RCL*, that would improve this situation if inserted into the current menu, in exchange of another recipe. From *RCL*, the algorithm chooses randomly and evaluates the existing menu to choose which item to extract. This is done by evaluating the impact of this extraction in the objective function, so the one that gives the greatest value is removed. This second approach can polish this candidate solutions.

In fact, when used independently, the first method offers results that are between 30 and 40% better than those obtained by the second approach. However, when used sequentially, the overall results are improved up to an additional 5%.

Note that, with the aim to introduce diversity within the same menu, a given plate cannot be suggested more than a fixed number of times, T. This rule has, as previously mentioned, a few exceptions, regarding bread, breakfast items and drinks. In

this work, this parameter was set to a frequency of three for the whole plan. This means that for a 15-day period, a feasible menu cannot suggest the same plate more than 3 times ($T = 3$).

It is necessary to refer to the uniqueness of the solutions. When we find the final array of solutions for the system of inequalities, these are unique menus that are solutions to the optimization problem, so they comply with every n-day period constraint and work properly daily, avoiding excessive repetition and energy imbalance. To understand how these menu plans are unique, it is due to explain when two (or more) menus are considered identical to each other. This occurs when both the overall frequency of all recipes and the daily structure (regardless of the ordination of the days) of every day are exactly the same.

Following this procedure, GRASP provides, in a short time, a reasonable set of solutions, i.e., complete menus that satisfy all the requirements established by the inequalities. However, one may wonder if the combination of these generated solutions would hide new ones. Because of this, as a second stage of the algorithm, we apply a Local Search Procedure (LSP) adapted to our MPP, which -originally- share some many similarities with genetic algorithms, that has been successfully used for large combinatorial problems. If *S* denotes the current set of complete menus generated using GRASP and *LS* the list of the local search procedures:

$$\text{LS} = \{\text{LocalSearch1,LocalSearch2,Shake}\}$$

- LocalSearch1: Given 2 different candidate menu plans (s_1, s_2), it interchanges either all the lunches or dinners between them, as described in Fig.3.

  If the solution is considered as a matrix, where the columns represent the dth day of the horizon plan and the *k*th row indicates the specific meal, this search

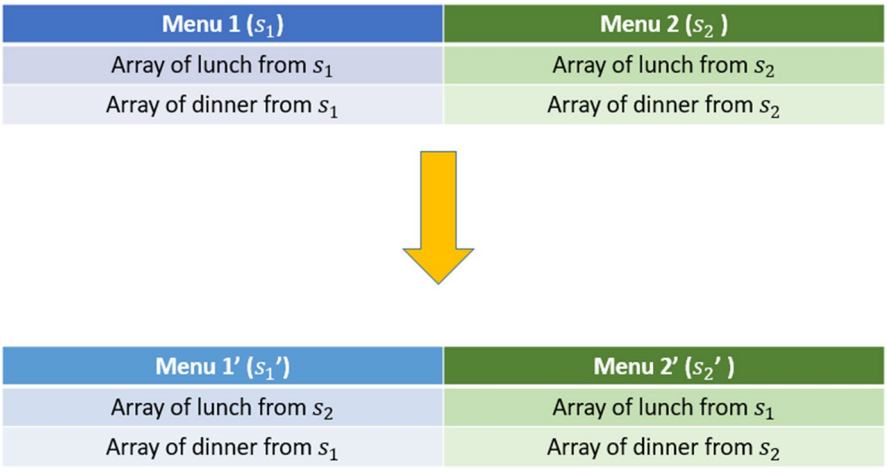

Fig. 3 Partial exchange given by type of intake

Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
...						
$Menu_1$ Lunch 1	$Menu_1$ Lunch 2	$Menu_1$ Lunch 3	$Menu_1$ Lunch 4	$Menu_1$ Lunch 5	$Menu_1$ Lunch 6	$Menu_1$ Lunch 7
...						
$Menu_1$ Supper 1	$Menu_1$ Supper 2	$Menu_1$ Supper 3	$Menu_1$ Supper 4	$Menu_1$ Supper 5	$Menu_1$ Supper 6	$Menu_1$ Supper 7
...						

Fig. 4 Hypothetical solution s_1. (Similar to hypothetical s_2)

Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
			...			
$Menu_2$ Lunch 1	$Menu_1$ Lunch 2	$Menu_2$ Lunch 3	$Menu_1$ Lunch 4	$Menu_1$ Lunch 5	$Menu_2$ Lunch 6	$Menu_1$ Lunch 7
			...			
$Menu_2$ Supper 1	$Menu_1$ Supper 2	$Menu_2$ Supper 3	$Menu_1$ Supper 4	$Menu_1$ Supper 5	$Menu_2$ Supper 6	$Menu_1$ Supper 7
			...			

Fig. 5 Introducing days from menu 2 in menu 1

studies if the menus generated by interchanging the rows $k = 6, 7$ and $k = 10$ are solution of the problem.

- LocalSearch2: Given 2 different solutions, s_1 and s_2, as detailed in Fig. 4 and any similar solution, it interchanges a random number of complete days between the two solutions. So, considering this particular solution, the procedure extracts a random number (R, R $\leq$ D) of days from one menu and from another and randomly exchange them.

  The following figure (Fig. 5) shows the introduction into menu s_1 of days from s_2. In this case, the structure of menu 1 for its days 2, 4, 5 and 7 remains intact. However, day 1, 3 and 6 come now from menu s_2. Similarly, there is another resulting menu from this combination that would consist on menu s_2 with days 1, 3 and 7 from menu s_1 inserted.

  where the red marks represent the days cut from s_1 and pasted into the structure of day s_2. Again, if the solution is considered as a matrix, where each row indicates the kth intake, this search studies if the menus generated by mixing a random number from s_1, c, and $D - c$ from s_2, of days (columns of the matrix) are solution of the problem.
- Shake: For a given menu plan, which is already a solution of the problem, this local search permutes randomly either the main recipes from lunch or dinner and evaluates if the daily constraints are satisfied. This approach ensures to maintain the value of the objective function in the shaken menu.

The strategy behind *LocalSearch*1 and *LocalSearch*2 is inspired on the philosophy of recombination, a feat present in genetic algorithm. It has been proved to be a good alternative to generate solutions from a pool. Hence, during this *LS* procedure, for any resulting menu whose evaluation is checked to be under 0.15, so although promising, it does not belong to the current set of solutions (S), will also be considered to combine. It means that: $S = \{s : f(s) < 0.15\}$. Now, for each solution $s \in S$, a local search from *LS* is applied while new solutions are found. However, if no new solutions are found for a given number of iterations (*maxIter*) the process continues with another pair of solutions. A general pseudocode for *LSP* is given by the following stages:

Algorithm 3 Step 2. Local Search Procedure.

procedure LOCAL SEARCH PROCEDURE(S, LS)
  Set $MAX = 10$;
  Set $nonImprov = 0$;
  $S_{def} = \emptyset$
  while $nonImprov \leq MAX$ **do**
    for each pair $s_1, s_2 \in S$ **do**
      $s' = LocalSearch1(s_1, s_2)$; ▷ $S = \{s : f(s) < 0.15\}$
      if $f(s') = 0$ **then**
        $S = \{s'\} \cup S$;
        $nonImprov = 0$;;
      else
        $nonImprov = nonImprov + 1$;
      end if
    end for
  end while
  $nonImprov = 0$;
  while $nonImprov \leq MAX$ **do**
    for each pair $s_1, s_2 \in S$ **do**
      $s' = LocalSearch2(s_1, s_2)$; ▷ $S = \{s : f(s) < 0.15\}$
      if $f(s') = 0$ **then**
        $S = \{s'\} \cup S$;
        $nonImprov = 0$;;
      else
        $nonImprov = nonImprov + 1$;
      end if
    end for
  end while
  $nonImprov = 0$;
  while $nonImprov \leq MAX$ **do**
    for each $s \in S$ **do**
      $s' = Shake(s)$;
      if s' satisfies Eq. 15 **then**
        $S_{def} = \{s'\} \cup S_{def}$;
        $nonImprov = 0$;;
      else
        $nonImprov = nonImprov + 1$;
      end if
    end for
  end while
  return S.

Note that for *LocalSearch*1 and *LocalSearch*2, given two different solutions, another 2 are generated, so s' represent either one or the other. The key would be to find, at least, one new solution. Additionally, in the view that preferences change, the pool of solutions can be easily modified to discard some menus that have a newly undesired condition or to add some previously discarded menus that are now acceptable. It is also important to mention that for *LocalSearch*1 and *LocalSearch*2, the daily condition given by Eq. 15, that does not allow to consume fish or meat for lunch and dinner at the same day, is not taken into account. This is why the algorithm includes shaking local search, which consists of a permutation of either the lunch meals or dinner, in order to comply with Eq. 15.

The running time that the algorithm requires to generate the forementioned menus depends on the number of recipes included in the reference database as well as the number of iterations considered. In Sect. 3.3, some tests justify that, regarding computational time, it is convenient to run a shorter number of times the GRASP step and leave the second step, the recombination LSP, to generate a denser set of solutions.

3.3 Parameters study

Now, in order to obtain the best results from GRASP, there are a few parameters that require to be adjusted. The parameter tuning phase is always a challenging part of any metaheuristic approach, so in our work we have tried to explore different combinations of all of them to arrive to a good set of solutions. These parameters also include iteration-limits and as-good-as-null limit to accept values in the objective function when launching each of the local search strategies. Thus, have provided a satisfactory amount of solutions. Due to a lack of benchmark instances, the tests have been run using our current data set.

So, to determine the best value of *constantRandom* for our study we have performed Parameter Tuning (El-Ghazali 2009). It is important to study what value should be considered to construct the solution, i.e., what *constantRandom* is more appropriate to set, in order to generate better complete menus during the initial construction stage of our proposal. In this context, several tests have been run assigning different values to *constantRandom*. Note that if *constantRandom* is 1, the construction phase is deterministic, as it will always include the best short-sighted possible option. However, it does not guarantee an optimal solution so, for better results, a random candidate will be chosen from a restricted candidate list (*RCL*), composed by a maximum of *constantRandom* elements. The larger *constantRandom*, the more diversity in the *RCL*. The following table shows the results to study the performance (objective function value and time) of the GRASP-construction phase:

In particular, Table 7 shows a statistical summary of the objective function values and the time required after running 1000 iterations of the constructive phase of GRASP, where the parameter *constantRandom* under study is equal to 5, 10, 15, 20, 25 and 30. Regarding these results, $constantRandom = 15$ is the most balanced, as it offers good results in the quality of the solutions and in the time employed.

Table 7 Statistical results to test constantRandom

Test-summary; f (& time)	ConstantRandom = 5	ConstantRandom = 10	ConstantRandom = 15	ConstantRandom = 20	ConstantRandom = 25	ConstantRandom = 30
MAX	7.3438 (47.9753)	8.7141 (15.2849)	4.7906 (15.8967)	3.1749 (14.1539)	3.2587 (15.5688)	4.1531 (18.5057)
MIN	0.0483 (5.1662)	0.0953 (14.4605)	0.0733 (1.0290)	0.1037 (0.9287)	0.1557 (14.4367)	0.1181 (14.4569)
VARIANCE	0.2406 (21.0587)	0.2122 (0.0245)	0.1419 (6.8037)	0.0804 (6.7943)	0.1224 (0.0178)	0.2262 (0.3423)
MEDIAN	0.3658 (14.7722)	0.3985 (14.6200)	0.3980 (14.5784)	0.4602 (14.5832)	0.5673 (14.5871)	0.6273 (15.1411)

In fact, it shows the value of f and the time employed per menu generated is displayed for a collection of different values of *constantRandom*. For example, with *constantRandom* set to 20, the minimum value of f was 0.1037 -very close to the feasible set- and the fastest a menu was generated was in 0.93 seconds. The median helps us see that, for the same value of *constantRandom*, 50% of the menus had an objective function value lower or equal than 46.02%. The variance and the maximum values can be similarly interpreted.

After considering a $constantRandom = 15$ for the constructive phase of GRASP, the procedure is followed by an improvement phase. In this phase, we continue the philosophy of considering a *RCL* with a fixed length (α) in order to reduce the space of search. Note that the candidate list is generated according to the local searches described at Sect. 3. Once again, we evaluate the basic statistical parameters -in order to choose the most appropriate value for α-, the number of feasible solutions found, the improvement rate and the time required to obtain such results for different values of α in a test of 1000 iterations. So, given the same set of constructed solutions from the constructive phase, the following table summarizes the test results and times required for different values of α:

Along the experiments, a total of 407 solutions have been found for values of α greater than 15. Then, regarding the Table 8, we observe just slightly differences between the values of the improvement ratio for each case. This table shows the improvement in the value of f (minimum, median and variance) and the times taken per menu improved. Maximum values are not considered because they are all 100%, showing that the best result regardless of the value of α is reducing the f value of a menu to 0. However, the *time (seconds)* required to reach the zero-value from the constructed solution looks more significant, so that we consider it as the criterion to determine the best α, which, in this case, is set to $\alpha = 15$.

Now, despite the good results obtained from GRASP, we decided to cut on the time to generate as many solutions as possible. As observed, each step of the algorithm takes a different amount of time to compute. The initial generation (*construction*) of random seeds spends about 3 seconds per menu -the exact time depends on the *randomConstant* (i.e. deterministic constructions are slightly slower)- and the amount of information we store, along the program run, about these seeds. Then, the improvement phase within GRASP takes, in average, 6.8 seconds. Therefore, if we reduce the number of initial iterations run and combine the results, the computational time will be also cut down. Now, the second step of the proposed algorithm tries to densify the current set of solutions, as detailed in Sect. 3, using a sequence of *Local Search Procedures*. The order of the neighborhoods to search is fixed, as established in Sect. 3:

1. *LocalSearch*1
2. *LocalSearch*2
3. *Shake*

In particular, *LocalSearch*1 and *LocalSearch*2 begin from the set of menus $S' = \{s : f(s) < 0.1\}$ where s represents a complete menu. Then, both apply

Table 8 Statistical analysis for the improvement and time of the set of solutions among different length of the candidate list for a 1000 iteration test

Improvement rate	$\alpha = 5$ (time, s)	$\alpha = 10$ (time, s)	$\alpha = 15$(time, s)	$\alpha = 20$(time, s)	$\alpha = 25$ (time, s)	$\alpha = 30$(time, s)
MIN	1.04% (5.1662)	1.04% (1.0860)	1.04% (1.0290)	1.04% (0.9287)	1.04% (1.0023)	1.04% (0.8929)
MEDIAN	99.18% (20.6712)	99.19% (8.1543)	99.20% (7.6984)	99.18% (7.6094)	99.20% (7.9678)	99.20% (7.7712)
VARIANCE	0.58% (36.6404)	0.40% (21.1810)	0.31% (17.4629)	0.31% (17.1237)	0.31% (17.4054)	0.31% (16.8821)

different techniques to combine pairs of menus from S and try to find new solutions to the system of inequalities in which we have transformed the MPP. Besides, an additional set is defined with those menus that satisfy the global requirements (Tables 4-6), but not the daily restrictions (Eq. 15-14). The latter are tackled using the *Shake* operator, which shuffle either the lunch or dinner main dishes -between different days- looking for compliance of the daily requirements. Setting the *constantRandom* to 15, the *numMirar* to 15, we check the number of initial seeds required to obtain a good set of solutions in a reasonable time. Actually, numIter $= \{10, 25, 50, 100\}$ iterations have been considered. For a larger number of iterations, the computation effort required is not worth it. Table 9 sums up the final experiment for each of these number of iterations considered:

From Table 9, we observe the contribution of each of the local search procedures in terms of the total non-repeated solutions found. In particular, the given information includes the number of solutions (non-repeated) generated that comply the daily requirements after each step, as well as the time required. For *numIter* set to 25, the number of solutions created with GRASP was 18, and it took 655 seconds. However, none of those 18 solutions were complying with the daily restrictions. After applying the *LocalSearch2*, out of the 122 solutions -generated so far- complies with

Table 9 Results of testing the number of iterations

Phase		GRASP	Local Search Procedure			Total
			LocalSearch1	LocalSearch2	Shake	
numIter = 10	Number of solutions	6	6	6		
	Complywith daily requirements			0	0	0
	(time,s)	249.15	8.74	1.05	0.68	10.47
	Ratio numSol/time		0.686514566	5.714829984	0	0
numIter = 25	Number of solutions	18	47	122		
	Comply with daily requirements			1	5	5
	(time,s)	655.82	46.87	23.93	9.81	80.61
	Ratio numSol/time		1.002668806	5.098757073	0.51	0.06
numIter = 50	Number of solutions	32	147	704		
	Comply with daily requirements			7	110	110
	(time,s)	1,051.04	187.32	212.56	53.59	453.48
	Ratio numSol/time		0.78	3.31	2.05	0.24
numIter = 100	Number of solutions	58	624	23614		
	Comply with daily requirements			139	1851	1851
	(time,s)	2,053.23	661.66	7,307.78	1,560.17	9,529.62
	Ratio numSol/time		0.94	3.23	1.19	0.19

the daily requirements. In the end, we are interested in this kind of solutions. That is why, in the total column, these are the only ones considered.

As observed, the time invested in the first phase (GRASP) increases with the number of iterations, which makes more difficult to consider more iterations. Moreover, as expected, the time required to run *LocalSearch1* and *LocalSearch2* depends on the number of initial iterations, due to the fact that they not only consider the set of solutions ($S = \{s : f(s) = 0\}$), but they also consider complete menus with a "good" f-value. It is greater as we increment the number of iterations and *LocalSearch2* becomes slower as more solutions are generated from *LocalSearch1*. This fact is observed comparing the values for the *ratio numSol/time* that informs about the number of solutions found per second at each step of the algorithm. Finally, *Shake* is a fast and efficient approach, applied in order to find solutions, within the current pool of complete menus, that satisfy the daily conditions.

4 Results

In this context, the number of complete menus in S, is not relevant since the algorithm searches for solutions that comply the daily requirements between those with an objective value of 0. This might be a key point of the methodology, as it enables us to find a larger set of solutions to our problem in a shorter time. The *ratio* of solutions found per unit of time shows a better performance with 50 iterations than with 100 iterations. Each column at *Number of Solutions*' rows indicates how many solutions have been found at the time; whereas rows *Comply with daily requirements* indicates the number of solutions that verify the daily conditions established by Eqs. 14-15. Also, calculating the ratio of number of new solutions found after shake, for 50 and 100 iterations, a better result is obtained for 50 iterations, despite a larger number of menus is found for 100 iterations. The test shows a small proportion of solutions that verify the daily conditions. Rows denoted as *time (seconds)* shows the seconds invested in the particular procedure, while *Total* only counts the time required for the Local Search Procedure phase.

From Table 9 we may observe:

- The more iterations, the more solutions found. However, the computational time also increases enormously. In particular, *LocalSearch2* takes a very long time, as it has to combine all the menus previously found.
- Considering the *Shake* local search highly contributes to find complete menus that comply the daily requirements in a very short time.
- The ratio informs about the number of solutions generated per second. This is an interesting measure to compare these experiments.
- The results generated with 50 iterations shows the best performance, despite it does not provide the larger set of solutions.

In fact, we can actually see how the algorithm is performing when a seed is randomly generated and then improved towards becoming a solution. This is shown in Fig. 6. The value of the objective function is seen after every change performed to

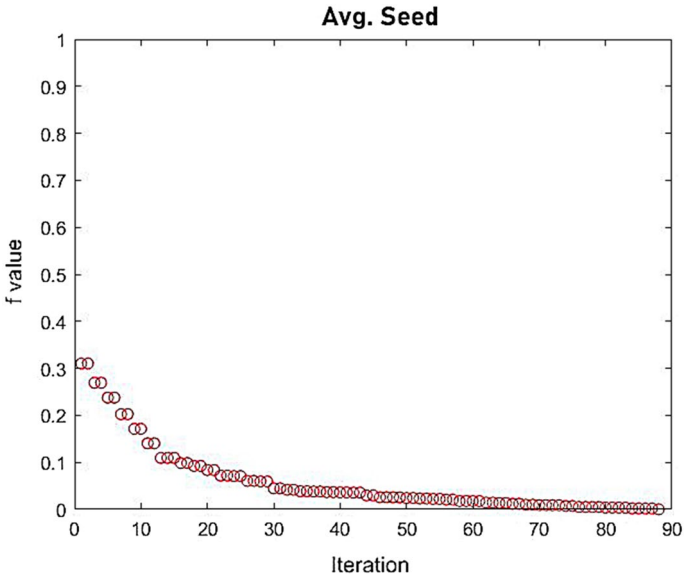

Fig. 6 Improvements of the average seed

the candidate menu that is being modified and evaluated. The average seed -between those that end up becoming solutions, starts with an f-value of 0.33 and after less than 90 movements, it is brought to the feasible region.

Then, our problem is solved both at a global level- building complete menus with a null f value- and at a local level -our solutions are those which, additionally, satisfy the daily conditions- as can be seen in Table 9. Therefore, our solution complies with every restriction at a global level -the whole 15 days-, and also respects the macro-nutrient balance in the daily caloric intake that is required while avoiding excessive repetition of the main ingredients.

Concerning the solutions, we have summarized them, in order to understand the performance of the algorithm and to see which plates and ingredients -the actual solutions are menu plans- are chosen more frequently and to which food groups they do belong (Fig. 7).

The results -taking into account that they are healthy nutritious menus- show a large consumption of Vegetables, Fruits, Dairy and Grains and a low intake of Sauces and Condiments and Sweets, which corresponds to the standards of the Mediterranean Diet. It suggests an average distribution of a menu as shown in Fig. 8.

This figure shows the distribution of the menu in Proteins, Vegetables and Fruits and Grains as a plate. The average result is not far from the Harvard Plate (Willett and Skerrett 2017). Note that, in this figure, "Proteins" enclose the consumption of Eggs, Dairy, Meat (Regular and Processed), Fish, Legumes, Nuts and Other vegetable proteins and to "Rest" belong -cooking- fats (mainly EVOO), Sweets and Condiments and Spices. Liquids are not taken into consideration for this figure. This shows the importance to keep a balance in a daily menu between the most common ingredients, but at the same time shows a simple way of making a healthy menu.

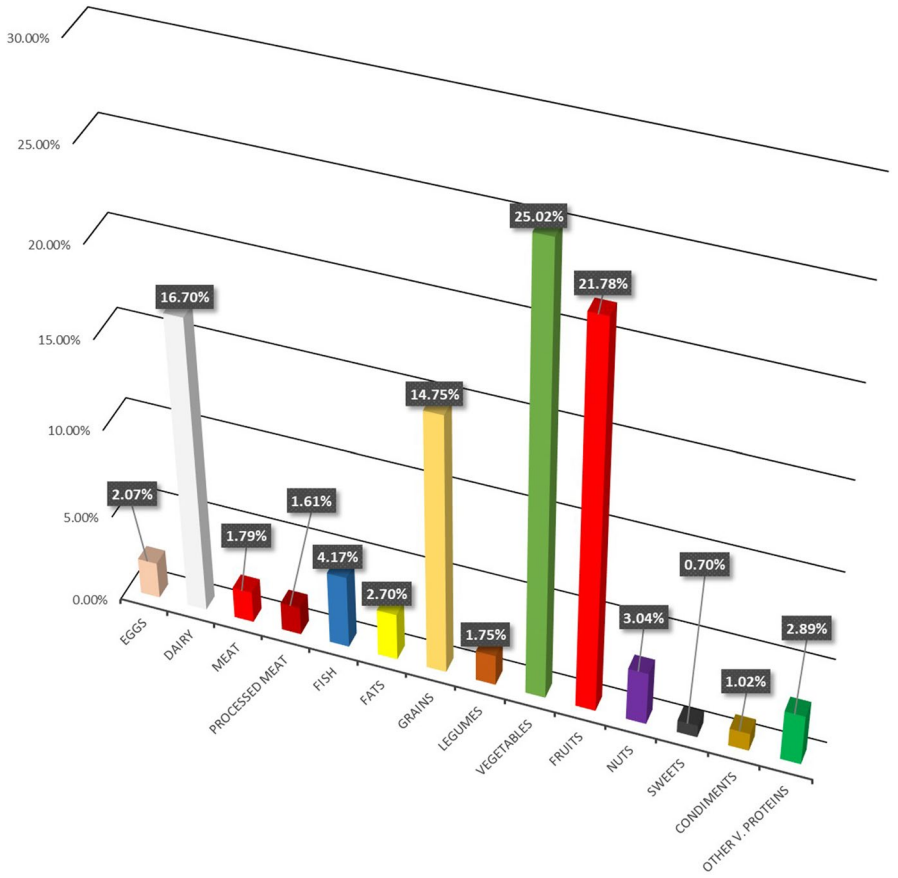

Fig. 7 Groups in the menu

With the appropriate amount of veggies and fruits, quality protein, grains and cooking fats, eating healthy is an easy habit.

The proportion of Proteins suggested along the solutions is shown in Fig. 9, where dairy products and fish constitutes more than 60% of the Proteins of the solutions menu. The Mediterranean standards usually give importance to fish over meat, and also to legumes and nuts as complementary sources of protein. Regarding dairy, usually is not seen as a main part of a proper Mediterranean diet, but its presence as another source of protein has been proven to be beneficial (Wade et al. 2018). Our solutions also point in that direction.

In particular, among actual recipes, the most repeated ones are shown in Fig. 10, but liquids and the bread that accompanies lunch have been left out.

According to the construction of the menu, which is based on the structure defined by Table 3, it makes sense that fruits or nuts are very present in this list. Fruits are included as a mandatory component in every breakfast, as well as nuts as a daily snack. There is also a very appealing dessert -custard- and spinach as an

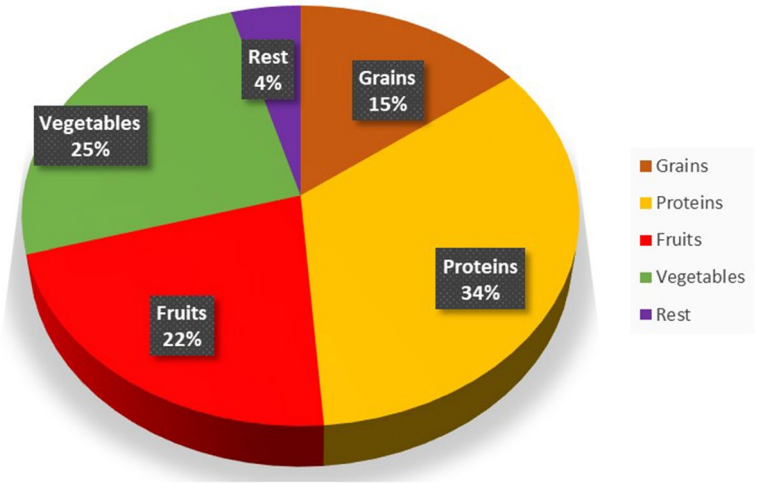

Fig. 8 Composition of the average menu

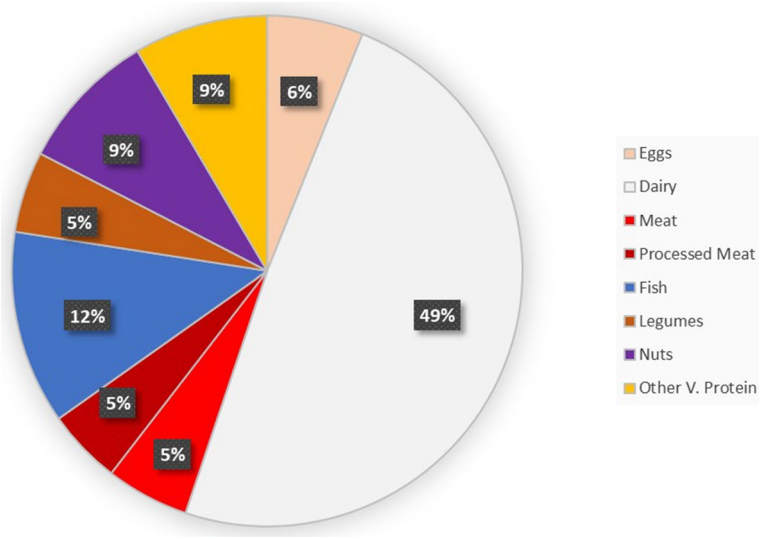

Fig. 9 Protein Distribution

ingredient appears in three recipes in the list, which shows its good qualities as a vegetable. The number of different drinks -not present in the list- is also reduced, so mostly water appears in every lunch and dinner, but also the occasional glass of soda, wine or beer. The inclusion of these two alcoholic drinks in an optimized menu can also seem conflicting. However, beer and wine are a part of an accepted Mediterranean Diet when consumed with moderation. As a consequence of their presence, the resulting menus can be perceived as more appealing and realistic.

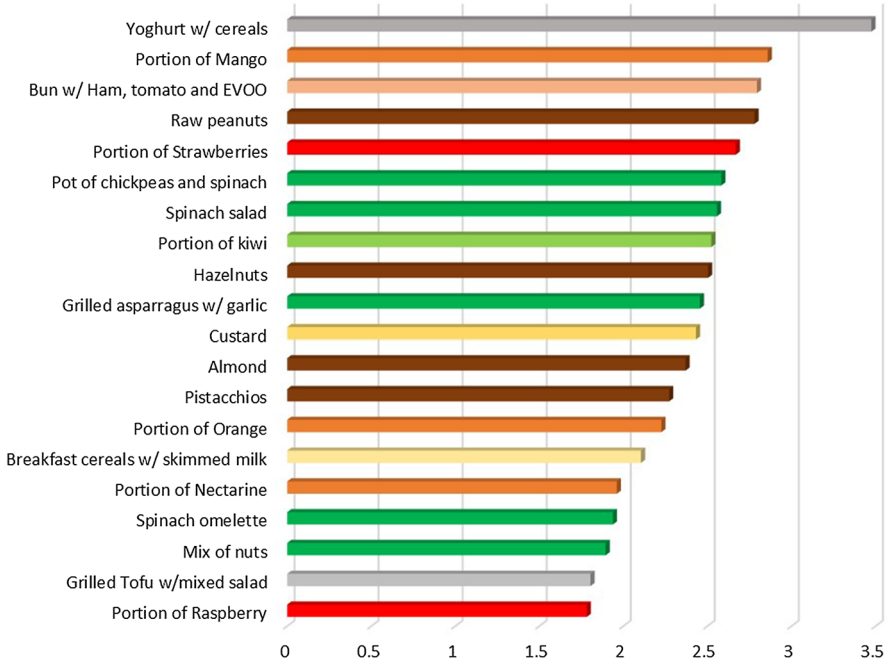

Fig. 10 Most repeated plates

It might be surprising to see a recipe with Tofu as the main ingredient in this list, before any with meat or fish. The authors have discussed the fact that vegetarian protein recipes are scarcer in the database, and that, together with the restriction to avoid repeating fish or meat for lunch and dinner -and the variety of recipes with them-, is promoting this particular dish. However, 'other vegetarian protein', the group to which Tofu belongs and that also contains tempeh or wheat gluten only accounts for 9% of all the protein intake.

It might be relevant to acknowledge that some new food items are being brought into our diets. In the case of Spain, this means that the Mediterranean Diet is being complemented with some new proteins, such as tofu or tempeh. However, this has implications about the new trends and food habits, and these results could be a reflection on these.

The results obtained, when applying the algorithm, depart from the presented structure (see Table 2) offering an enhanced pool of D-days menus. Our algorithm is able to generate consistent appealing menus. As an example, we present a part -the first 4 days- of a 15-day complete, nutritious and affordable menu that has been extracted from the pool of solutions:

Table 10 is (or shows) the first part of a random -but representative enough- biweekly menu generated by our algorithm. Not only is it a complete menu, so affordability and nutrition are granted, it also avoids extreme repetition and, most importantly, it is appealing -palatable-. The allowance of moderate amounts of alcohol -when permitted- is also a notable feat. Alcohol, or the presence of any specific

ingredient or nutrient, can be easily avoided -by introducing a new inequality, in case we want to forbid it.

Regarding the violation of the constraints, there are a few interesting points. First, when running the algorithm, the program generated multiple menus. Among those menus, there were a number of solutions, because they complied with every constraint, but there were also menus generated -in the search for feasibility- that could not become solutions. These menus found difficulties in the satisfaction of some of the constraints.

Given the set of recipes considered, there were some constraints that were more frequently unsatisfied. The cost was almost never a problem -we were generous with the allowance-, as were Iron, Magnesium, Potassium, Vitamin B12 -no vegan menus-, Vitamin C or Fiber. However, the incorrect proportion of energy from sugar and from carbohydrates was present responsible for between 70 and 90% of failed menus. The proper amount of vegetables failed to be met in 40–60% of unfeasible menus, the same percentage as Vitamin E, Calcium and Vitamin B9 -Folacin-. Lastly, in around 5–10% of these menus, Cholesterol, Sodium, Vitamin A, Vitamin D and Fat goals were not met.

5 Conclusions

To sum up, the main contribution of this work is that it proceeds from a formal mathematical model to use a hybrid algorithm to solve the Menu Planning Problem (MPP). This consists of defining a set of conditions, mathematically expressed in terms of a system of inequalities to actually find a considerable number of feasible menus. We minimize the distance to the feasible region to find these solutions that are, in fact, healthy realistic menus. Specifically, in our work, we are able to find solutions to this system, a mathematical programming problem with no restrictions is defined, whose objective function is specified in order to find the set of complete menus that do not violates any conditions. To attain that goal, it is important to generate a large list of recipes and establish a structure to follow.

Besides, the MPP is a combinatorial problem and an exact method is not applicable. Hence, a simple metaheuristic is designed and adapted to this particular problem. The algorithm proposed consists of 2 phases: first, we generate a set of complete menus using GRASP, trying to find the combination of recipes that, evaluating the objective function, its value is zero. In this context, different parameters of GRASP have been tested for either the construction and the improvement stages.

The performance of the algorithm has been satisfactory, as it has been able to obtain an ample set of feasible solutions. The average time employed in every step of the process have been recorded. Although the seed generation has not been able to provide feasible solutions, the GRASP applied to such seeds has had a success rate averaging 60% at pushing them towards the feasible region. The number of found unique feasible -only at global level- candidates with our database of recipes and ingredients is huge -in the thousands for some specific parameter combinations-.

Table 10 4 days example of an obtained solution

Consumption pattern	Day 1	Day 2	Day 3	Day 4
Juice or piece of fruit	Banana	Apple and pear juice	Portion of strawberries	Kiwi
Hot Beverage	Coffee w/ semi-skimmed milk no sugar	Cocoa w/ soy milk	Coffee w/ semi-skimmed milk no sugar	Coffee w/ semi-skimmed milk no sugar
Breakfast dish	Cereals w/ yoghurt	Bread and EVOO	Bun with ham, EVOO and tomato	Ham & Cheese Sandwich
Small bun of bread	Regular	Whole	Whole	Regular
Cold drink	Water	Wine glass	Water	Water
First dish	Garlic & almond cold soup	Salad	Veggie pie	Veggie pot
Main dish	Stew w/ rice	Grilled eggplant	Filled pepper	Crumbed fillet w/ ham and cheese and rice
Dessert	Grapes	Melon	Fruit w/ yoghurt	Plum sauce
Cold drink	Water	Water	water	20cl beer
Main dish	Spinach salad	Cod à Brás	Grilled wheat gluten w/ rice	Pepper and onion salad
Dessert	Yoghurt w/ fruit	Kiwi	Kiwi	Mango
Snacks	Raw peanuts	Salted fried peanuts	Mix of nuts	Raw pistachios

This methodology can be used, therefore, as a ground step for this kind of optimization problems, to explore and expand a feasible set. A considerable number of solutions have been found using the proposed algorithm. Similar models could benefit from using it.

Within this work, in a model with more than 40,000 binary variables and a few tens of constraints, we have been able to find and enrich the feasible set, having then feasible menus that are affordable and nutritious for a number D of given days. We firmly believe that complementary lines of work could include the introduction of new food related constraints linked to different approaches to specific diets, considering individual likes and dislikes, allowing a wide selection of menus according to personal preferences.

Funding Open Access granted by Universidad de Málaga / CBUA. This work has been partially supported by the Spanish *Ministerio de Ciencia, Innovación y Universidades *(MCIU/AEI/FEDER, UE) with grant ref PID2019-104263RBC42; and Junta de Andalucía with grant refs. P18-RT-1566, (contract ref CI-21-228) UMA18-FEDERJA- 065.

References

Aggarwal M, Grady A, Desai D, Hartog K, Correa L, Ostfeld RJ, Freeman AM, McMacken M, Gianos E, Reddy K, Batiste C, Wenger C, Blankstein R, Williams K, Allen K, Seifried RM, Aspry K, Barnard ND (2020) Successful implementation of healthful nutrition initiatives into hospitals. Am J Med 133(1):19–25. https://doi.org/10.1016/j.amjmed.2019.08.019

Bach-Faig A, Fuentes-Bol C, Ramos D, Carrasco J, Roman B, Bertomeu IF, Cristià E, Geleva D, Serra-Majem L (2010) The mediterranean diet in spain: adherence trends during the past two decades using the mediterranean adequacy index. Public Health Nutr 14(4):622–628

Balintfy JL (1964) Menu planning by computer. Commun ACM 7(4):255–259. https://doi.org/10.1145/364005.364087

Barosh L, Friel S, Engelhardt K, Chan L (2014) The cost of a healthy and sustainable diet - who can afford it? Aust N Z J Public Health 38(1):7–12. https://doi.org/10.1111/1753-6405.12158

Benvenuti L, Santis De A (2020) Making a sustainable diet acceptable: an emerging programming model with applications to schools and nursing homes menus. Front Nutr 7

Benvenuti L, De Santis A, Santesarti F, Tocca L (2016) An optimal plan for food consumption with minimal environmental impact: the case of school lunch menus. J Clean Prod 129:704–713

Benvenuti L, De Santis A, Di Sero A, Franco N (2019) Concurrent economic and environmental impacts of food consumption: are low emissions diets affordable? J Clean Prod 236:117645

Benvenuti L, De Santis A, Cacchione P (2021) Multi-indicator design and assessment of sustainable diet plans. J Clean Prod 313:127699

Bröckling U, Krasmann S, Lemke T (2010) Governmentality: current issues and future challenges. Routledge, New York, https://doi.org/10.4324/9780203846476

Burlingame B, Dernini S (2011) Sustainable diets: the mediterranean diet as an example. Public Health Nutr 14(12A):2285–7. https://doi.org/10.1017/s1368980011002527

Buttriss JL, Briend A, Darmon N, Ferguson EL, Maillot M, Lluch A (2014) Diet modelling: how it can inform the development of dietary recommendations and public health policy. Nutr Bull 39(1):115–125. https://doi.org/10.1111/nbu.12076

Dantzig GB (1949) Programming of interdependent activities: ii mathematical model. Econom, J Econom Soc 17(3,4):200–211

Eckstein E (1970) Communication to the editor–is the "diet problem" identical to the "menu planning problem"? Manag Sci 16(9):527–528. https://doi.org/10.1287/mnsc.16.9.527

Eckstein EF (1967) Menu planning by computer: the random approach to planning for consumer acceptability and nutritional needs. Diss Abstr: B 51(6):529–533

El-Ghazali T (2009) Common Concepts for Metaheuristics, chapter 1, pp 1–86. Wiley, Ltd, 2009. ISBN 9780470496916. https://doi.org/10.1002/9780470496916.ch1

Feo TA, Resende MGC (1995) Greedy randomized adaptive search procedures. J Global Optim 6(2):109–133. https://doi.org/10.1007/bf01096763

Ferrari M, Benvenuti L, Rossi L, De Santis A, Sette S, Martone D, Piccinelli R, Le Donne C, Leclercq C, Turrini A (2020) Could dietary goals and climate change mitigation be achieved through optimized diet? the experience of modeling the national food consumption data in italy. Front Nutr 7:48

Funabiki N, Taniguchi S, Matsushima Y, Nakanishi T (2011) A proposal of a menu planning algorithm for two-phase cooking by busy persons. In: 2011 international conference on complex, intelligent, and software intensive systems, pp 668–673, 2011. https://doi.org/10.1109/CISIS.2011.112

Gazan R, Brouzes CMC, Vieux F, Maillot M, Lluch A, Darmon N (2018) Mathematical optimization to explore tomorrow's sustainable diets: a narrative review. Adv Nutr 9(5):602–616. https://doi.org/10.1093/advances/nmy049

Gerdessen JC, de Vries JHM (2015) Diet models with linear goal programming: impact of achievement functions. Eur J Clin Nutr 69(11):1272–1278. https://doi.org/10.1038/ejcn.2015.56

Germani A, Vitiello V, Giusti AM, Pinto A, Donini LM, del Balzo V (2014) Environmental and economic sustainability of the mediterranean diet. Int J Food Sci Nutr 65(8):1008–1012. https://doi.org/10.3109/09637486.2014.945152

Guala S, Marenco J (2020) Scheduling weekly menus in a hospital with integer programming techniques. Revis Investig Oper 41(1):67–79

Gue RL (1969) A reformulation of the menu planning problem. A I I E Trans 1(2):146–149. https://doi.org/10.1080/05695556908974426

Hernández M, Gómez T, Delgado-Antequera L, Caballero R (2019) Using multiobjective optimization models to establish healthy diets in Spain following mediterranean standards. Op Res. https://doi.org/10.1007/s12351-019-00499-9

Hernandez-Ocana B, Chavez-Bosquez O, Hernandez-Torruco J, Canul-Reich J, Pozos-Parra P (2018) Bacterial foraging optimization algorithm for menu planning. IEEE Access 6:8619–8629. https://doi.org/10.1109/access.2018.2794198

Ivancic A, Kanellopoulos A, Bloemhof-Ruwaard J, Geleijnse M (2018) Towards modelling sharp diets, based on nutritional adequacy, sustainability metrics and population diversity parameters. Technical report, SUSFANS

Kanellopoulos A, Gerdessen JC, Ivancic A, Geleijnse JM, Bloemhof-Ruwaard JM, vant Veer P (2020) Designing healthier and acceptable diets using data envelopment analysis. Public Health Nutr 23(13):2290–2302. https://doi.org/10.1017/S1368980019004774

Lancaster LM (1992) The history of the application of mathematical programming to menu planning. Eur J Oper Res 57(3):339–347. https://doi.org/10.1016/0377-2217(92)90345-a

Lartey A (2019) Sustainable healthy diets: guiding principles. Food and Agriculture Organization of the United Nations World Health Organization, Rome. 9789251318751

Leung P, Wanitprapha K, Quinn LA (1995) A recipe-based, diet-planning modelling system. Br J Nutr 74(2):151–162. https://doi.org/10.1079/bjn19950119

Maillot M, Vieux F, Amiot MJ, Darmon N (2009) Individual diet modeling translates nutrient recommendations into realistic and individual-specific food choices. Am J Clin Nutr 91(2):421–430. https://doi.org/10.3945/ajcn.2009.28426

Marling CR, Petot GJ, Sterling LS (1999) Integrating case-based and rule-based reasoning to meet multiple design constraints. Comput Intell 15(3):308–332. https://doi.org/10.1111/0824-7935.00095

Marrero A, Segredo E, León C, Segura C (2020) A memetic decomposition-based multi-objective evolutionary algorithm applied to a constrained menu planning problem. Mathematics 8(11):1960. https://doi.org/10.3390/math8111960

Merrigan K, Griffin T, Wilde P, Robien K, Goldberg J, Dietz W (2015) Designing a sustainable diet. Science 350(6257):165–166. https://doi.org/10.1126/science.aab2031

Mertens E, vant Veer P, Hiddink GJ, Steijns JM, Kuijsten A (2016) Operationalising the health aspects of sustainable diets: a review. Public Health Nutr 20(4):739–757. https://doi.org/10.1017/s1368980016002664

Michel M, Burbidge A (2019) Nutrition in the digital age - how digital tools can help to solve the personalized nutrition conundrum. Trends Food Sci Technol 90:194–200. https://doi.org/10.1016/j.tifs.2019.02.018

Moraes L, Fadel J, Castillo A, Casper D, Tricarico J, Kebreab E (2015) Modeling the trade-off between diet costs and methane emissions: a goal programming approach. J Dairy Sci 98(8):5557–5571. https://doi.org/10.3168/jds.2014-9138

Moreira RP, Wanner E, Martins FVC, Sarubbi JF (2018) An evolutionary mono-objective approach for solving the menu planning problem. In: 2018 IEEE congress on evolutionary computation (CEC), pp 1–8, https://doi.org/10.1109/CEC.2018.8477888

Moreiras O, Carbajal A, Cabrera L, Cuadrado C (2017) Tablas de Composición de Alimentos: Guía de Prácticas. Grupo Anaya, ISBN 978-84-368-3623-3

Nugent R, Bertram MY, Jan S, Niessen LW, Sassi F, Jamison DT, Pier EG, Beaglehole R (2018) Investing in non-communicable disease prevention and management to advance the sustainable development goals. The Lancet (London, England) 391(10134):2029–2035. https://doi.org/10.1016/S0140-6736(18)30667-6 (**ISSN 0140-6736**)

Petot GJ, Marling C, Sterling L (1998) An artificial intelligence system for computer-assisted menu planning. J Am Diet Assoc 98(9):1009–1014. https://doi.org/10.1016/s0002-8223(98)00231-4

Pichugina O (2020) Diet-menu problem modelling and applications. In: 2020 IEEE 2nd international conference on system analysis intelligent computing (SAIC), pp 1–5, https://doi.org/10.1109/SAIC51296.2020.9239149

Prais M, Ribeiro CC (1999) Parameter variation in grasp implementations. In: Extended abstracts of the third metaheuristics international conference, pp 375–380

Prais M, Ribeiro CC (2000) Parameter variation in grasp procedures. Investig Op 9(1):1–20

Resende M, Ribeiro CC (2003) Greedy randomized adaptive search procedures, pp 219–249

Rutten M, Achterbosch TJ, de Boer IJ, Cuaresma JC, Geleijnse JM, Havlík P, Heckelei T, Ingram J, Leip A, Marette S, van Meijl H, Soler L-G, Swinnen J, vant Veer P, Vervoort J, Zimmermann A, Zimmermann KL, Zurek M (2018) Metrics, models and foresight for european sustainable food and nutrition security: the vision of the SUSFANS project. Agric Syst 163:45–57. https://doi.org/10.1016/j.agsy.2016.10.014

Segredo E, Miranda G, Ramos JM, León C, Rodríguez-León C (2020) Schoolthy: automatic menu planner for healthy and balanced school meals. IEEE Access 8:113200–113218. https://doi.org/10.1109/ACCESS.2020.3003067

Shrimpton R, Rokx C (2012) The double burden of malnutrition : a review of global evidence. Health, Nutrition and Population Discussion Paper, URL https://openknowledge.worldbank.org/handle/10986/27417

Smith VE (1959) Linear programming models for the determination of palatable human diets. J Farm Econ 41(2):272–283. https://doi.org/10.2307/1235154

Steuer RE, Choo E-U (1983) An interactive weighted tchebycheff procedure for multiple objective programming. Math Program 26(3):326–344

Stigler GJ (1945) The cost of subsistence. J Farm Econ 27(2):303–314. https://doi.org/10.2307/1231810

Sufahani S, Ismail Z (2014) A new menu planning model for malaysian secondary schools using optimization approach. Appl Math Sci, 8:7511–7518, https://doi.org/10.12988/ams.2014.49725

Syahputra MF, Felicia V, Rahmat RF, Budiarto R (2017) Scheduling diet for diabetes mellitus patients using genetic algorithm. J Phys: Conf Ser 801:012033

Toledo RY, Alzahrani AA, Martinez L (2019) A food recommender system considering nutritional information and user preferences. IEEE Access 7:96695–96711. https://doi.org/10.1109/access.2019.2929413

Trichopoulou A, Costacou T, Bamia C, Trichopoulos D (2003) Adherence to a mediterranean diet and survival in a greek population. N Engl J Med 348(26):2599–2608

Wade AT, Davis CR, Dyer KA, Hodgson JM, Woodman RJ, Murphy KJ (2018) A mediterranean diet supplemented with dairy foods improves markers of cardiovascular risk: results from the meddairy randomized controlled trial. Am J Clin Nutr 108(6):1166–1182

Waxman A (2004) Who global strategy on diet, physical activity and health. Food Nutr Bull 25(3):292–302. https://doi.org/10.1177/156482650402500310

Willett WC, Skerrett PJ (2017) Eat, drink, and be healthy: the Harvard Medical School guide to healthy eating. Simon and Schuster

Willett WC, Sacks F, Trichopoulou A, Drescher G, Ferro-Luzzi A, Helsing E, Trichopoulos D (1995) Mediterranean diet pyramid: a cultural model for healthy eating. Am J Clin Nutr 61(6 Suppl):1402S-1406S