Journal of Food Composition and Analysis 22 (2009) 414–420

Contents lists available at ScienceDirect

Journal of Food Composition and Analysis

journal homepage: www.elsevier.com/locate/jfca

Original Article

Computer-based dietary menu planning

Barbara Koroušić Seljak [*]

Computer Systems Department, Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 18 October 2007
Received in revised form 26 January 2009
Accepted 19 February 2009

Keywords:
Optimal nutrition
Dietary recommendations and guidelines
Computer-based menu planning
Nutrient balancing
Evolutionary computation
Multi-objective and multi-constraint optimization
Linear programming
Pareto-optimal solutions
Food composition

ABSTRACT

In this paper, we introduce a computer-based method for menu planning, which applies evolutionary computation. First, we formalize the *n*-day menu-planning problem, decomposing it into several sub-problems at the daily-menu and meal-planning level. We reduce the problem to a multi-dimensional knapsack problem. Then, we define an evolutionary algorithm that quickly finds a diverse set of feasible solutions (i.e. optimal menus) with the optimum objective functions' values, without examining all the possibilities. As the problem is constrained, infeasible solutions need to be repaired in order to direct the "evolution" towards the feasible regions. We present greedy repairing methods that slightly differ at the global level and the sub-problems' levels. At the meal-planning level, we couple repairing with linear programming to balance infeasible meals. We conclude the paper with the presentation of empirical results, which showed that the evolutionary method may outperform a human. A computer was able to find the Pareto-optimal front of 21-day menus with respect to a dietary advice in equal or less time than a human professional, who designed a daily menu. However, the human factor is still important in the last stage, when a solution has to be selected from the Pareto front.

1. Introduction

Increasing incidence of chronic diseases (CDs) in modern societies requires a shift from adequate towards *optimal nutrition*, not only providing us with the energy and nutrients but also contributing to the well-being and health. Optimal nutrition, as part of a healthy lifestyle, is an important factor in the prevention of CDs and may also have a therapeutic potential.

Today, many recommendations and guidelines on optimal nutrition, based on results of advanced research methods and tailored to the needs of a society, are available. They consider the current knowledge of the relationship between our immune system, aetiology of CDs and health status. However, their implementation in practice is difficult for several reasons, including the complexity of dietary recommendations and guidelines and limited health literacy that may lead to misunderstanding. Dollahite et al. (1995) found that professionally designed menus published in diet manuals may fail to meet all recommendations and guidelines.

In this paper, we propose a computer-based method for planning optimal menus with respect to agreed evidence-based dietary recommendations and guidelines. This method assists a human in fitting regular menus to new health paradigms.

The paper is organized as follows: Section 2 presents a brief survey of computer-based methods for menu planning and an introduction to evolutionary computation and multi-objective optimization; Section 3 provides a formulation of the menu-planning problem; Section 4 describes an evolutionary algorithm for menu planning; and Section 5 presents empirical results and concluding remarks.

2. Computer-based methods for menu planning

Menu planning is an art that one often learns by a costly heuristic or trial-and-error process. A well-trained professional can cope with the complexity of regular menu planning, but as soon as one attempts to meet the needs of diverse groups, control costs and quality, and schedule production tightly for maximum utilization of labour and equipment time, the probability of success diminishes. Therefore, computer-based methods, which facilitate the routine decisions in menu planning, are useful.

2.1. History

Four decades ago, the need for computer-based methods for menu planning was recognized. As a result, the quality of information and data necessary for computerization of the menu-planning process became available to many institutions and its apparent feasibility thus increased. In 1964, Balintfy (Balintfy, 1964; Eckstein, 1983) applied *linear programming*

[*] Tel.: +386 1 477 33 63; fax: +386 1 477 38 82.
E-mail address: barbara.korousic@ijs.si.

0889-1575/$ – see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jfca.2009.02.006

techniques to build the first computer-based menu planner, which optimized menus for nutritional adequacy and budgeted food cost. Brown (1966) developed primitive techniques for controlling the palatability of individual non-selective menus using *random selection techniques*. A year later, these techniques were adopted by Eckstein (1967) to satisfy menus with several constraints, comprising: cost, color, texture, shape, calories, variety and acceptability by the target population.

In the 1970s, interest in computer applications in menu planning appeared to wane; lack of funding and inadequate software components were contributing causes.

In the 1990s, when the field of artificial intelligence was revived, the menu-planning problem became popular again. Two types of menu planners, *case-based* and *rule-based*, were implemented. One of them was JULIA (Hinrichs, 1992), an interactive menu planner that was used to plan meals to satisfy a group of guests, despite conflicting food preferences and evolving constraints. Ganeshan and Farmer (1995) implemented a Prolog catering system. Marling et al. (1999) designed a menu-planning tool for individuals, taking dietary requirements and personal preferences into account. That tool integrated case-based and rule-based reasoning to meet multiple constraints.

A comprehensive review of the use of optimization techniques based on linear and nonlinear programming was given by Darmon et al. (2002).

2.2. Evolutionary computation

The computer-based method for menu planning we have recently proposed (Koroušić Seljak, 2004) is based on *evolutionary computation*.

In computer science, evolutionary computation is a subfield of artificial intelligence that involves *numerical* and *combinatorical optimization* (CO) problems. Evolutionary computation uses iterative progress, such as growth or development in a population of potential problem solutions. The field comprises many techniques, mostly involving metaheuristic[1] optimization algorithms, such as evolutionary algorithms (EAs) and swarm intelligence (Korošec and Šilc, 2008). These techniques rely on analogies to natural processes; some of them have been inspired by biological mechanisms of evolution. The first ideas were developed in the 1960s by Holland (1961) and Fogel (Fogel and Owens, 1966), and have already reached a stage of some maturity (Michalewicz, 1996).

In recent years, numerous algorithms taking inspiration from nature have also been proposed to handle *continuous* optimization problems: real-coded genetic algorithms using some specific operators, evolution strategies using Gaussian mutations with adaptive or self-adaptive update strategies, and differential evolution, to name a few.

As real-world optimization problems may involve objectives, constraints and parameters, which constantly change with time, *dynamic* consideration using evolutionary computation methods have also raised a lot of interest within the last few years. For these dynamic and uncertain optimization problems the objective of the evolutionary algorithm is no longer to simply locate the global optimum solution, but to continuously track the optimum in dynamic environments, or to find a robust solution that operates optimally in the presence of uncertainties.

[1] A metaheuristic is a heuristic method for solving a very general class of computational problems by combining user-given black-box procedures — usually heuristics themselves — in, it is hoped, an efficient way. The name combines the Greek prefix "meta" ("beyond", here in the sense of "higher level") and "heuristic" (from ευρισκειν, heuriskein, "to find").

Optimization is a procedure of finding and comparing feasible solutions until no better solution can be found. Solutions, which in our case are healthy menus, are termed as follows:

- good or bad in terms of multiple conflicting *objectives*, such as: cost, quality of ingredients, aesthetic standards or other factors; and
- feasible if they satisfy all the problem *constraints* that are defined by dietary recommendations and guidelines.

While classical deterministic optimization methods can at best find one solution in one simulation run, evolutionary techniques are more efficient in finding multiple trade-off optimal solutions in a single simulation run. These solutions have a wide range of values for each objective representing the multi-dimensional *Pareto-optimal front*,[2] requiring an additional decision-making activity for choosing a single solution from the front.

3. Menu-planning model

As today's computers have few limitations, satisfiable menus can be automatically or semi-automatically generated by efficient software (SW) techniques, but only if the menu-planning process is well defined.

Mathematically, menu planning can be reduced to a *multi-dimensional* (i.e. *multi-constrained and multi-objective*) *knapsack problem* (*MDKP*), which is a widely studied CO problem that has many direct counterparts (Garey and Johnson, 1979).

3.1. Formulation of the MDKP

Given foods of different values and volumes, the MDKP is to find the most valuable combination of foods that fits in a knapsack of fixed volumes. *Values* are defined subjectively with respect to food quality, cost and aesthetic parameters (comprising taste, consistency, color, temperature, shape and method of preparation). Knapsack *volumes* are defined by dietary recommendations and guidelines.

3.2. Complexity of the MDKP

MDKP is easy to formulate, yet its decision problem is *NP-complete*[3] (Garey and Johnson, 1979). In complexity theory, the NP-complete problems are the most difficult problems, which cannot be solved by exact SW techniques in deterministic

[2] For a given system, the *pareto-optimal front* is a set of non-dominated solutions, i.e. solutions that cannot be improved upon without hurting at least one of the objectives. The term is named after Vilfredo Pareto, an Italian economist.

[3] In computational complexity theory, the complexity class *NP-complete* (standing for nondeterministic polynomial time) is a class of problems having two properties:

- Any given solution to the problem can be *verified* quickly (in polynomial time); the set of problems with this property is called NP.
- If the problem can be *solved* quickly (in polynomial time), then so can every problem in NP.

Although any given solution to such a problem can be verified quickly, there is no known efficient way to locate a solution in the first place. Indeed, the most notable characteristic of NP-complete problems is that no fast solution to them is known. That is, the time required to solve the problem using any currently known algorithm increases very quickly as the size of the problem grows. As a result, the time required to solve even moderately large versions of many of these problems easily reaches into billions or trillions of years, using any amount of computing power available today. As a consequence, determining whether or not it is possible to solve these problems quickly is one of the principal unsolved problems in computer science today.

polynomial time but require time that is superpolynomial in the input size.

The knapsack values and the volumes are linear, but highly complex, because they are weakly correlated. As there are at least two optimal solutions that are not indifferent to each other, the problem is *multimodal*.

Another difficulty is that foods are selected from a food composition database (FCDB), which consists of several thousand items having tens of composition parameters. As a consequence, the decision space contains a large set of potential solutions to the menu-planning problem. Moreover, the problem landscape defined by the decision and the objective space contains several peaks.

However, there exist *heuristic SW techniques*, such as evolutionary computation techniques, that work "reasonably well" on many problem's instances, but for which there is no proof that they are both always fast and always produce a good result.

Comprehensive reviews of multi-constrained 0-1 knapsack problems, presenting a subset of MDKPs, and associated heuristic algorithms was given by Chu and Beasley (1998) and by Ishibuchi and Kaige (2003).

4. Evolutionary method for menu planning

We applied the state-of-the-art evolutionary algorithm (EA) *NSGA-II* (Elitist Non-Dominated Sorting Genetic Algorithm) (Deb, 2001) in a multi-level way (Gunawan et al., 2003) to solve the MDKP of menu planning. The main idea behind the method is to develop healthy meals and daily menus independently, guiding the optimization to overall Pareto-optimal *n*-day menus (Fig. 1). All objectives are treated as equally important. The decision on the best compromise to be chosen among adequate Pareto-optimal solutions is made after the search.

At the meal-planning level we solve three sub-problems (for planning meals that are composed of a breakfast and a morning snack, a lunch and an afternoon snack, and a dinner and a light snack before bedtime, respectively), and at the daily-menu planning level we solve five sub-problems (for daily menus with the main meal consisting of red meat, white meat, fish, soya/legumes, or eggs/curd, respectively).

4.1. Some basic definitions

In general, NSGA-II is a multi-objective EA that can be characterized by the use of three ideas: Pareto-dominance-based "fitness" evaluation, diversity maintenance, and elitism.

The *algorithm* comprises the following steps:

1. *Initialize "population" of potential solutions*
2. *Evaluate "fitness" of "individuals"*
3. *Estimate feasibility of individuals*
4. *Repair infeasible individuals*
5. *Selection:*
   - *Non-dominated sorting*
   - *Individual comparison*
6. *Recombination: Combine traits of "parents"*
7. *Mutation: Random walk around an individual*
8. *Evaluate "offspring" solutions*
9. *Replacement: Best among parents and offspring* (Fig. 2)
10. *Return to Step 5 or terminate*

The *non-dominated sorting* procedure identifies the best non-dominated set (i.e. a set of solutions that are not dominated by any individual in the population), discards them from the population temporarily, identifies the next best non-dominated set, and continues till all the solutions are classified.

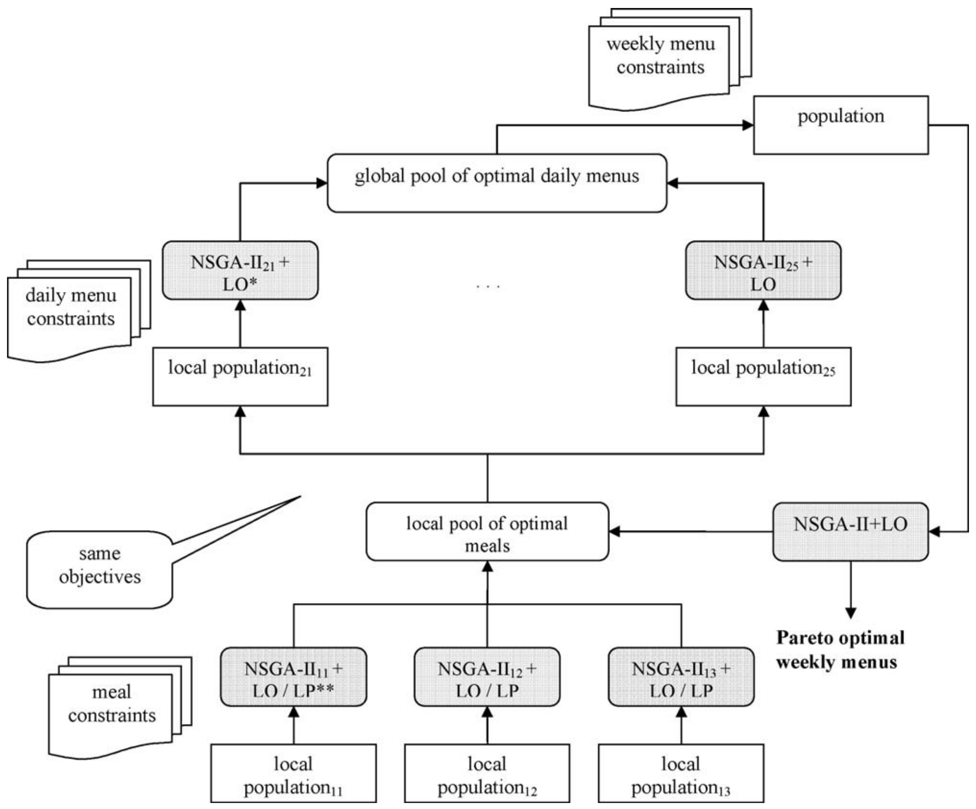

Fig. 1. Schematic of the evolutionary method for menu planning. *LO – local optimization (greedy repair) and **LP – linear programming.

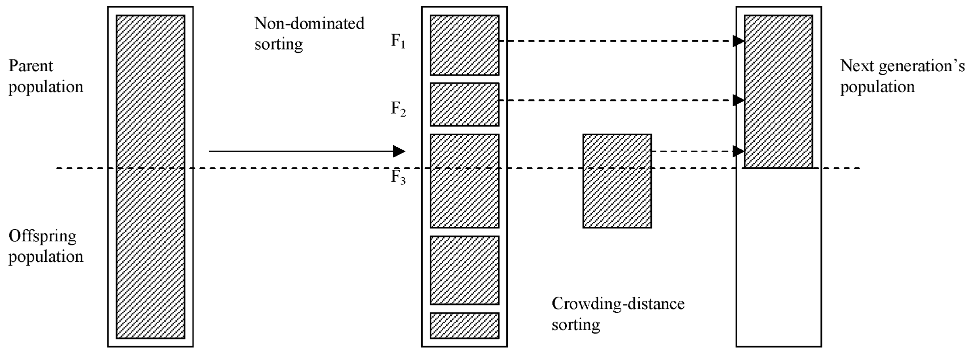

Fig. 2. Elitist replacement in NSGA-II.

The *crowding-distance sorting* procedure assigns each potential solution a crowding distance (i.e. an average distance from its nearest neighbours on either side of the solution along each of the objectives of the problem). A particular solution is more crowded than another solution if its front density in the neighborhood is higher.

During selection, NSGA-II uses a *crowded comparison* operator, which takes into consideration both the non-domination rank of an individual in the population and its crowding distance: i.e. non-dominated solutions are preferred over dominated solutions, but between two solutions with the same non-domination rank, the one that resides in the less crowded region is preferred.

4.1.1. Representation structures

In order to apply NSGA-II to the problem of menu planning, we first encode potential solutions of the n-day menu-planning problem and its sub-problems (of the daily menu and meal planning) by *integer-valued coding*. In our representation:

- at the n-day menu level, a "chromosome" contains n integer data, $n \in N$, $n \geq 2$, carrying the information about n daily menus: $[d_1, d_2, \ldots, d_n]$;
- at the daily-menu level, a "chromosome" contains m integer data, $m \in N$, $m \geq 3$ carrying the information about m basic and not composite meals: $[o_1, o_2, \ldots, o_m]$;
- at the meal level, a "chromosome" is formed of j pairs (c_i, x_i), where c_i denotes the FCDB code[4] of a food item i and x_i its quantity expressed in grams: $[(c_1, x_1, (c_2, x_2), \ldots, (c_j, x_j)]$. By default, the number of pairs j varies between 1 and 10, depending on the number of meal courses.

4.1.2. Initialization

The algorithm starts the "evolution" from an initial "population" of either random potential solutions or solutions (n-day menus) known from experience. The population's size remains constant over all "generations".

The menu-planning sub-problems at the daily and the meal level operate on local populations. Initially, the local population at the daily-menu level is filled with decomposed n-day menus from the global population, and the local population at the meal level is filled with decomposed daily menus from the daily-menu population.

Beside the global and the local populations, we use additional pools of potential meal and daily-menu solutions (Fig. 1) that have a function of an archive of the union of solutions generated by each sub-problem. Initially, the pools are empty.

[4] FCDBs normally consist of composition data for foods ordered by major food groups and subgroups (e.g. plain yogurt made of skim milk and plain yogurt made of whole milk have consecutive food codes).

4.1.3. Fitness

Each potential solution is evaluated as good or bad in terms of objectives using its fitness values. These are non-negative integer numbers calculated by the following *objective functions*:

$$f_k(\vec{x}) = \frac{1}{\sum_{i=1}^{n} v_{ik} x_i},\ k = 1, 2,\ f_3(\vec{x}) = \sum_{i=1}^{n} v_{i3} x_i,$$

$$f_4(\vec{x}) = \sum_{i=1}^{l} \lambda_i f_{4,i}(\vec{x}),\ \ \lambda_i \geq 0 \wedge \sum_{i=1}^{l} \lambda_i = 1,\ \ f_{4,l}(\vec{x}) = \left(\sum_{j=1}^{n_{a_l}} \left| \sum_{i=1}^{n} h_{lj}(x_i) - \frac{\sum_{i=1}^{n} h(x_i)}{n_{a_l}} \right| \right) - \sum_{i=1}^{l} h(x_i), \tag{1}$$

$$h_{lj}(x_i) = \begin{cases} 0, \text{ if } x_i = 0 \\ 1, \text{ if } x_i > 0 \wedge v_{il} = j \end{cases},\ \ h(x_i) = \begin{cases} 0, \text{ if } x_i = 0 \\ 1, \text{ otherwise} \end{cases},\ x_i \in N \cup \{0\},\ i = 1, 2, \ldots, n,\ l = 1, 2, \ldots 6,$$

where x_i denotes the quantity of the food item i (expressed in grams), v_{i1} and v_{i2} its functionality and quality in the season, respectively, λ_i the scalarization weight associated with the aesthetic parameter i, v_{i3}, v_{i4}, v_{i5}, v_{i6}, v_{i7}, v_{i8} and v_{i9} the cost, the taste, the consistency, the color, the temperature, the shape, and the method of preparation, respectively, and n_{a_l} the number of possibilities for the lth aestetic parameter. In order to reduce the number of objective functions, we apply the normalized weighted sum *scalarization technique* for assessment of the meal's or menu's variety.

The aim of EA at the global and the local levels is to *minimize* the objective functions of (1).

4.1.4. Feasibility

We estimate feasibility of a potential solution by *constraint functions* that differ at the global and the local levels.

At the *meal* level, the constraints are the least restrictive:

- The energy provided by a meal is limited by a lower and an upper bound:

$$g_3(\vec{x}) = \sum_{i=1}^{N_C} \frac{\omega_{iE} x_i}{100} \geq 0.9E,\ g_4(\vec{x}) = \sum_{i=1}^{N_C} \frac{\omega_{iE} x_i}{100} \leq 1.1E, \tag{2}$$

  where ω_{iE} denotes the number of calories in 100 g of the food item i, x_i the quantity of the item i expressed in grams, and E the recommended caloric value for the meal.
- The meal's energy density is limited both upwards and downwards:

$$g_5(\vec{x}) = \frac{\sum_{i=1}^{N_C} \omega_{iE} x_i}{\sum_{i=1}^{N_C} x_i} \geq 0.5E,\ g_6(\vec{x}) = \frac{\sum_{i=1}^{N_C} \omega_{iE} x_i}{\sum_{i=1}^{N_C} x_i} \geq 1.5E. \tag{3}$$

Table 1
Parameters of NSGA-II.

Parameter	n-Day menu planning	Daily-menu planning	Meal planning
String length	7	5	10
Population size	100	100	100
Pool size	–	500	300
Crossover type	Two-point[a]/Simulated binary (SBX) (Deb, 2001; Deb et al., 2007)[b]		
Crossover probability	0.7[a]/0.9[b]		
Mutation type	Linear descending mutation[c]		
Mutation probability	0.14–0.01	0.20–0.01	0.10–0.01
Probability of mutating a code or a quantity	–	–	0.5
Selection type	Binary tournament selection (based on the crowded comparison and feasibility operators)		
No. of iterations	250	250	250

[a] For codes.
[b] For quantities.
[c] Either on a code or on a quantity.

- The macronutrients are balanced:

$$g_7(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iP} 4x_i \geq b_{P_l} E, g_8(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iP} 4x_i \leq b_{Pu} E,$$
$$g_9(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iL} 9x_i \geq b_{L_l} E, g_{10}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iL} 9x_i \leq b_{L_u} E, \quad (4)$$
$$g_{11}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iC} 4x_i \geq b_{C_l} E, g_{12}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iC} 4x_i \leq b_{C_u} E,$$

where ω_{iP}, ω_{iL}, ω_{iC} denote the quantity of proteins, lipids and carbohydrates, respectively, in 100 g of the food item i. Because the quantities are expressed in grams, conversion factors[5] are required to attain to calories. $b_{P_l}, b_{Pu}, b_{L_l}, b_{L_u}, b_{C_l}, b_{C_u}$ denote the lower and the upper percentage of energy supplied by proteins, lipids and carbohydrates, respectively.
- Each food quantity (truncated by e_i) is limited by its original portion size:

$$g_1(\vec{x}) = \lfloor \frac{x_i}{e_i} \rfloor e_i \geq 0.25 P_i, g_2(\vec{x}) = \lfloor \frac{x_i}{e_i} \rfloor e_i \leq 2P_i. \quad (5)$$

At the *daily-menu* level, there are additional constraints that need to be satisfied:

- Simple sugars account for only 10% or less of RDI for the energy:

$$g_{13}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iS} 4x_i \leq 0.1 E_d, \quad (6)$$

where E_d denotes RDI for energy.
- The saturated fatty acids' intake is equal to or less than 10% of RDI for the energy:

$$g_{14}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iF} 9x_i \leq 0.1 E_d. \quad (7)$$

- RDI for the dietary fiber is *DF* grams per 1000-calorie energy intake and should not exceed 40 g:

$$g_{15}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iV} x_i \geq DF \frac{E_d}{1000}, g_{16}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iV} x_i \leq 40. \quad (8)$$

- The lower and the upper bounds for daily sodium intake are set at Na_l and Na_u, respectively:

$$g_{17}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iNa} x_i \geq Na_l, \; g_{18}(\vec{x}) = \sum_{i=1}^{N_C} \omega_{iNa} x_i \leq Na_u. \quad (9)$$

[5] We apply the conversion factors 4 for proteins and carbohydrates, and 9 for lipids.

At the *n-day menu* level, beside the meal and the daily-menu constraints, a chromosome presenting a n-day menu has to satisfy all the remaining constraints for nutrients, such as cholesterol, monounsaturated fatty acids, omega-3 and omega-6 polyunsaturated fatty acids, trans-fatty acids, water-soluble and fat-soluble vitamins, water, major minerals, and trace minerals, to be termed a feasible solution. Formal definitions of these constraints are similar to that of Eqs. (5) or (8).

4.1.5. Methods for repairing infeasible individuals

EAs have originally been developed for unconstrained problems. The most common way of incorporating constraints into EAs has been the use of a penalty function. Due to difficulties associated with the penalty methods, we apply a *repair method*[6] to handle constraints by NSGA-II:

- at the meal level, we first replace in each infeasible meal those courses that mostly contribute to the violation of constraints with similar but more appropriate ones (e.g. we replace beef broth with vegetable soup if there is a lack of fiber in a meal), and then convert infeasible solutions into feasible solutions using a deterministic local optimization procedure of *linear programming* (LP in Fig. 1). This procedure, based on the *simplex method* (Bhatti, 2000), refines the quantities of foods to satisfy the meal sub-problem constraints.
- at the daily-menu and the n-day menu level, we repair infeasible individuals by replacing critical meals with more appropriate ones. Critical meals are those that do not satisfy the constraints on the major food groups (i.e. breads, cereal, rice, and pasta/vegetables/fruits/milk, yogurt, and cheese/meat, poultry, fish, beans, eggs, and nuts/fats, oils, and sweets). Here, we use the problem-specific knowledge, considering a recommendation that (i) a daily menu has to be composed of a certain *number of food units* from each major food group and (ii) an n-day menu has to include a *diverse set of foods* from the major food groups. There may be limitations on frequency of red meat, fish, potatoes, etc.

For this aim, we apply the *Lamarckian repair scheme*, in which replacements of critical meals are used to generate new "offspring" (Ishibuchi et al., 2005).

The replacement of meal or menu elements that mostly contribute to the violation of constraints requires a prior sorting of elements. We apply the inverse non-dominated sorting,

[6] The design of the penalty function may greatly affect the algorithms ability to explore the feasible regions of the search space. Repair functions, on the other hand, restrict the search to the feasible regions. However, theoretical results recently published by He and Zhou (2007) confirmed observations from experiments that EAs using repairing infeasible solutions are better than those using penalizing infeasible functions at the finding a feasible solution.

Table 2
An optimal menu for patients with chronic kidney disease developed by using the evolutionary algorithm (EA) method (Rotovnik Kozjek et al., 2008).

	Female chronic patients	Male chronic patients
Average body weight/ height in Slovenia (kg/cm)	61.2/164.9	70.2/177
Recommended caloric value (RCV) (MJ/kcal)	8.8/2100	10/2400
	Calculated quantity value (g/ml)	
	Female chronic patients	Male chronic patients
Breakfast (25% of the RCV)		
Low-protein bread	175/Prot[a]: 2.9 g, AC: 2.9 g, EAC: 1 g, P: 56.1 mg, K: 106.4 mg	175/Prot: 2.9 g, AC: 2.9 g, EAC: 1 g, P: 56.1 mg, K: 106.4 mg
Grilled pepper	100/Prot: 0.9 g, AC: 0.3 g, EAC: 0.3 g, P: 13.7 mg, K: 117.6 mg	100/Prot: 0.9 g, AC: 0.3 g, EAC: 0.3 g, P: 13.7 mg, K: 117.6 mg
Olive oil	10/Prot: 0 g, AC: 0 g, EAC: 0 g, P: 0 mg, K: 0.1 mg	20/Prot: 0 g, AC: 0 g, EAC: 0 g, P: 0 mg, K: 0.2 mg
Blueberries	150/Prot: 0.9 g, AC: 0 g, EAC: 0 g, P: 19.5 mg, K: 117 mg	170/Prot: 1 g, AC: 0 g, EAC: 0 g, P: 22.1 mg, K: 132.6 mg
Sugar	5/Prot: 0 g, AC: 0 g, EAC: 0 g, P: 0 mg, K: 0.1 mg	10/Prot: 0 g, AC: 0 g, EAC: 0 g, P: 0 mg, K: 0.2 mg
Morning snack (15% of the RCV)		
Millet, cooked in water, with cream	100/Prot: 3.5 g, AC: 3.7 g, EAC: 1.3 g, P: 97.9 mg, K: 47.9 mg	110/Prot: 3.9 g, AC: 4.1 g, EAC: 1.5 g, P: 107.7 mg, K: 52.7 mg
Diet margarine	5/Prot: 0 g, AC: 0 g, EAC: 0 g, P: 0 mg, K: 0 mg	10/Prot: 0 g, AC: 0 g, EAC: 0 g, P: 0 mg, K: 0 mg
Stewed pears	100/Prot: 0.3 g, AC: 0 g, EAC: 0 g, P: 8 mg, K: 65 mg	120/Prot: 0.3 g, AC: 0 g, EAC: 0 g, P: 9.6 mg, K: 78 mg
Lunch (30% of the RCV)		
Stewed rice	210/Prot: 4.1 g, AC: 0 g, EAC: 0 g, P: 75.6 mg, K: 65.1 mg	240/Prot: 4.7 g, AC: 0 g, EAC: 0 g, P: 86.4 mg, K: 74.4 mg
Braised veal in vegetable sauce	210/Prot: 9.3 g, AC: 0.2 g, EAC: 0.1 g, P: 115.6 mg, K: 408.8 mg	240/Prot: 10.6 g, AC: 0.2 g, EAC: 0.1 g, P: 132.1 mg, K: 467.2 mg
Cucumber salad	170/Prot: 0.9 g, AC: 0.3 g, EAC: 0.2 g, P: 23.3 mg, K: 256.7 mg	190/Prot: 1 g, AC: 0.3 g, EAC: 0.2 g, P: 26.1 mg, K: 286.9 mg
Muffin	90/Prot: 3.6 g, AC: 4.1 g, EAC: 1.5 g, P: 99.8 mg, K: 86.8 mg	90/Prot: 3.6 g, AC: 4.1 g, EAC: 1.5 g, P: 99.8 mg, K: 86.8 mg
Afternoon snack (10% of the RCV)		
Pancake	90/Prot: 2 g, AC: 2.1 g, EAC: 0.9 g, P: 47.5 mg, K: 76.2 mg	90/Prot: 2 g, AC: 2.1 g, EAC: 0.9 g, P: 47.5 mg, K: 76.2 mg
Honey	20/Prot: 0.1 g, AC: 0 g, EAC: 0 g, P: 1 mg, K: 9 mg	30/Prot: 0.1 g, AC: 0 g, EAC: 0 g, P: 1.5 mg, K: 13.5 mg
Yeast	4/Prot: 1.9 g, AC: 1 g, EAC: 0.8 g, P: 76 mg, K: 56.4 mg	5/Prot: 2.4 g, AC: 1.2 g, EAC: 1 g, P: 95 mg, K: 70.5 mg
Dinner (20% of the RCV)	Calculated quantity value (g/ml)	
Low-protein bread	140/Prot: 2.3 g, AC: 2.3 g, EAC: 0.8 g, P: 44.9 mg, K: 85.1 mg	175/Prot: 2.9 g, AC: 2.9 g, EAC: 1 g, P: 56.1 mg, K: 106.4 mg
Sour cream	40/Prot: 1.2 g, AC: 1.3 g, EAC: 0.6 g, P: 34 mg, K: 52.8 mg	50/Prot: 1.6 g, AC: 1.7 g, EAC: 0.7 g, P: 42.5 mg, K: 66 mg
Cooked asparagus	100/Prot: 1.7 g, AC: 0 g, EAC: 0 g, P: 37 mg, K: 136 mg	120/Prot: 2 g, AC: 0 g, EAC: 0 g, P: 44.4 mg, K: 163.2 mg
Green salad	70/Prot: 0.7 g, AC: 0.3 g, EAC: 0.2 g, P: 13.8 mg, K: 109.1 mg	80/Prot: 0.8 g, AC: 0.3 g, EAC: 0.3 g, P: 15.8 mg, K: 124.6 mg
Cooked white of an egg	5/Prot: 0.6 g, AC: 0.5 g, EAC: 0.2 g, P: 1.1 mg, K: 7.7 mg	5/Prot: 0.6 g, AC: 0.5 g, EAC: 0.2 g, P: 1.1 mg, K: 7.7 mg
Caloric value (MJ/kcal)	8.5/2013	10/2373
kcal/kg TT	33	34
Nutritional value	Calculated value (the D–A–CH recommended values)	
Protein (g/kg of body weight)	0.6 (0.55–0.6)	0.59 (0.55–0.6)
Amino acids (g)	19.1	20.7
Essential amino acids[b] (% of all the amino acids)	40	40
Essential amino acids (g/kg TT)	0.12	0.12
Branched-chained amino acids (%)	20.5	20.7
Isoleucine (g)	1 (0.6)	1.1 (0.7)
Leucine (g)	1.7 (0.9)	1.9 (1)
Valine (g)	1.2 (0.6)	1.3 (0.7)
Conditionally essential aminoacids[c] (% of all the amino acids)	13	13
Fats		
Percentage of energy (%)	29	32
Saturated fatty acids (%)	7.1 (<10)	7.1 (<10)
Monounsaturated fatty acids (%)	9 (>10)	10.8 (>10)
Polyunsaturated fatty acids (%)	6.5 (3–7)	6.3 (3–7)
Carbohydrates		
percentage of energy (%)	63.7	61.2
Total dietary fiber (g)	27 (25 oz. >30)	30 (24 oz. >30)
Water-soluble vitamins		
Vitamin C (mg)	187 (110)	200 (100)
Vitamin B6 (mg)	1.1 (1.3)	1.2 (1.5)
Phonolic acid (μg equivalent)	334 (431)	385 (352)
P (mg)	765 (<1000)	858 (<1000)
K (mg)	1804 (<2000)	2031 (<2000)
Zn (mg)	4 (7.6)	5 (10)
Se (μg)	27 (30-70)	30 (30–70)
Na (mg)	2051 (1800–2500)	2282 (1800–2500)
I from the iodized cooking salt (μg)	55 (220)	60 (200)
Water (ml)	1337 (<2000)	1503 (<2000)

[a] Prot—protein; AC—amino acids; EAC—essential amino acids; P—phosphorus; K—potassium.
[b] Isoleucine, leucine, lysine, methionine, phenylalanine, threonine, tryptophan and valine.
[c] Arginine, cysteine, glycine and tyrosine.

meaning that the *replacing procedure* (LO in Fig. 1) starts replacing the elements from the last front and ends replacing the elements from the Pareto-optimal front. As a matter of fact, because the procedure is *greedy*, the replacement is stopped as soon as the first improvement is achieved. In this way, the cost of repair is minimized.

Parameter settings: The parameters of the NSGA-II procedure, including the types of operators, are specified in Table 1. Detailed description of the operators is beyond the scope of the paper.

4.1.6. Time complexity

The basic operations and their worst-case complexities are as follows:

1. Non-dominated sorting of individuals is $O(M(2N)^2)$ and of all infeasible individuals' elements is $O(M(2L_C)^2)$, where M denotes the number of objectives (1), N the population's size, and L_C the length of a chromosome;
2. Crowding distance assignment is $O(M(2N)\log(2N))$;
3. Sorting on the crowding comparison operator is $O2N\log(2N)$; and
4. Although it is known that specific variants of the simplex method require exponential time in the worst case (Megiddo, 1987), our empirical results have shown that in our case the LP complexity is $O({N_{Co}}^2)$, where N_{Co} denotes the number of constraints on the meal-planning sub-problem.

The overall complexity of EA for menu planning that solves L sub-problems including the global problem of n-day menu planning is $O(LMN^2)$, which is governed by the non-dominated sorting part of the algorithm.

5. Empirical results and discussion

The evolutionary method for menu planning has already been applied to the redesign of sample menus for children, workers and patients with special nutrition needs, which have recently been published in the Slovene Guidelines for Child Nutrition (Ribič Hlastan et al., 2008), the Slovene Guidelines for Workplace Nutrition (Pokorn et al., 2008), and the Slovene Recommendations for Clinical Nutrition and Nutrition for Elderly People in Care Homes (Rotovnik Kozjek et al., 2008), respectively.

The evolutionary method outperformed professionals in terms of time and quality. While it takes an experienced nutritionist or dietician from 30 min to 3 h to manually plan a daily menu for an individual or a group of individuals, a computer (1.7 GHz PentiumM, 512 MB RAM, Apache/PHP) needed from several minutes to a couple of hours to design a well-converged and well-distributed Pareto-optimal front of well-balanced optimal 21-day menus, depending on the complexity of the problem instance. The analysis showed a large percentage of infeasible solutions (an average 86%) in all sub-problems that we managed to reduce by LO and LP (an average to 34%).

We selected the most appropriate solutions from the Pareto-optimal fronts according to decisions made by a human after the search.

In Table 2, an example of a daily menu for patients with chronic kidney disease selected from the Pareto-optimal front developed by the evolutionary method is given. The menu was optimized upon a collection of daily menus for healthy adults, considering the following specific constraints (Rotovnik Kozjek et al., 2008):

- Recommended value for protein: 0.55–0.6 g/kg of body weight;
- Recommended value for potassium (K): 1500–2000 mg.

The caloric and nutritional values were calculated using food composition data from the national FCDB for meat and meat products, the Souci–Fachmann–Kraut FCDB (Scherz and Senser, 2000) and the USDA FCDB, and a recipe calculation method recommended by INFOODS that applies weight yield and retention factors published by Bognár (2002).

Acknowledgments

The author gratefully acknowledges the contributions of Prof. Dražigost Pokorn, MD, and the support provided by the Slovenian Ministry of Health during the course of this work.

References

Balintify, J.L., 1964. Menu planning by computer. Commun. ACM 7 (4), 255–259.
Bhatti, M.A., 2000. Practical Optimization Methods. Springer-Verlag, New York.
Bognár, A., 2002. Tables of weight yield of food and retention factors of food constituents for the calculation of nutrition composition of cooked foods (dishes). Bundesforschungsanstalt für Ernährung, Karlsruhe.
Brown, R.M., 1966. Automated menu planning. M.S. Thesis. Kansas State University, Manhattan, KS, USA.
Chu, P.C., Beasley, J.E., 1998. A genetic algorithm for the multidimensional knapsack problem. J. Heurist. 4, 63–86.
Darmon, N., Ferguson, E., Briend, A., 2002. Linear and nonlinear programming to optimize the nutrient density of a population's diet: an example based on diets of preschool children in rural Malawi. Am. J. Clin. Nutr. 75, 245–253.
Deb, K., 2001. Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Ltd..
Deb, K., Karthik, S., Okabe, T., 2007. Self-adaptive simulated binary crossover for real-parameter optimization. In: Proceedings of GECCO'07, London, UK, pp. 1187–1194.
Dollahite, J., Franklin, D., McNew, R., 1995. Problems encountered in meeting recommended dietary allowances for menus designed according to the dietary guidelines for Americans. J. Am. Diet. Assoc. 95, 341–347.
Eckstein, E.F., 1967. Menu planning by computer: the random approach. J. Am. Diet. Assoc. 51, 529–533.
Eckstein, E.F., 1983. Menu Planning, third ed. The AVI Publishing Company, Inc..
Fogel, L.J., Owens, A.J., 1966. Artificial Intelligence Through Simulated Evolution. John Wiley & Sons, Ltd., New York.
Ganeshan, K., Farmer, J., 1995. Menu planning system for a large catering corporation. In: Proceedings of the 3rd International Conference on the Practical Application of Prolog, Paris, France.
Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman.
Gunawan, S., Farhang-Mehr, A., Azarm, S., 2003. Multi-level multi-objective genetic algorithm using entropy to preserve diversity. In: Fonseca, C., Fleming, P.J., Zitler, E., Deb, K., Thiele, L. (Eds.), Lecture Notes in Computer Science. Springer-Verlag, Berlin, ISSN 0302-9743.
He, J., Zhou, Y., 2007. A comparison of GAs using penalizing infeasible solutions and repairing infeasible solutions on restrictive capacity knapsack problem. In: Proceedings of GECCO'07, London, UK, p. 1518.
Hinrichs, T., 1992. Problem Solving in Open Worlds: A Case Study in Design. Lawrence Erlbaum Associates, Hillsdale, NJ.
Holland, J.H., 1961. A Logical Theory of Adaptive Systems–Informally Described. The University of Michigan, Ann Arbor, MI, pp. 1–5.
Ishibuchi, I., Kaige, S., 2003. Comparison of multiobjective memetic algorithms on 0/1 knapsack problems. In: Proceedings of GECCO'03, Chicago, USA.
Ishibuchi, I., Kaige, S., Narukawa, K., 2005. Comparison between Lamarckian and Baldwinian repair on multiobjective 0/1 knapsack problems. In: Coello Coello, C.A., et al. (Eds.), EMO'05, LNCS 3410. Springer-Verlag, Berlin, pp. 370–385.
Korošec, P., Šilc, J., 2008. Comput. Informat. 27 (3) 377–402.
Koroušić Seljak, B., 2004. Evolutionary balancing of healthy meals. Informatica 28, 359–364.
Marling, C.R., Petot, G.J., Sterling, L.S., 1999. Integrating case-based and rule-based reasoning to meet multiple design constraints. Comput. Intel. 15 (3), 308–332.
Megiddo, N., 1987. On the complexity of linear programming. In: Bewley, T.F. (Ed.), Advances in Economic Theory. Fifth World Congress, Cambridge University Press, pp. 225–268.
Michalewicz, Z., 1996. Genetic Algorithms + Data Structures = Evolution Programs, third revised and extended ed. Springer.
Ribič Hlastan, C., Maučec Zakotnik, J., Koroušić Seljak, B., Pokorn, D., 2008. The Slovene Guidelines for Child Nutrition—A Practical Course. Cindi Slovenia, The National Educational Institute and Ministry of Health of the Republic of Slovenia (in Slovene).
Pokorn, D., Maučec Zakotnik, J., Močnik Bončina, U., Koroušić Seljak, B., 2008. The Slovene Guidelines for Workplace Nutrition. Cindi Slovenia and Ministry of Health of the Republic of Slovenia (in Slovene).
Rotovnik Kozjek, N., et al. (Eds.), 2008. The Slovene Recommendations for Clinical Nutrition and Nutrition for Elderly People in Care Homes. Oncological Institute Ljubljana, Slovenia, and Ministry of Health of the Republic of Slovenia (in Slovene).
Scherz, H., Senser, F., 2000. Souci-Fachmann-Kraut, Die Zusammensetzung der Lebensmittel, Nährwert-Tabellen [German tras. of Souci-Fachmann-Kraut, Food Composition and Nutrition Tables, sixth ed.]. Medpharm Scientific Publishers, Stuttgart.