Latifa.dekhici@univ-usto.dz

Operational Research

Practical Work Sheet n. 2 . Local Search and Metaheuristics

The goal is to optimize the selection and distribution of N food items across three meals a day (breakfast, lunch, and dinner), while satisfying nutritional requirements, adhering to constraints on food categories, and maintaining feasibility within given limits. Some items are not recommended to be eaten at dinner.

Sets

- I : Set of all N food items, $I = \{1,2,\ldots,N\}$.
- J: set of M nutrients (calories, protein, vitamin A...)
- K : Set of 3 meals, $K = \{1,2,3\}(1 =$ breakfast, $2 =$ lunch, $3 =$ dinner).
- G : Set of food categories, $G = \{1,2,\ldots,G_{\max}\}$. Drink, Principal item, Dessert

Variables

- $x_{i,k}$: Quantity of food item i assigned to meal k (in 100 g).

Parameters

cat (i,g) : Binary parameter indicating whether food i belongs to category g.

A_{ij} : Amount of nutrient j in 100 g of food i (e.g, protein, vitamins).

B_j : Daily requirement for nutrient $j, j \in \{1,\ldots,M\}$.

c_i : Cost per 100 g of food i.

$q_{\max}(i)$: Maximum quantity (in 100 g) of food i allowed per day.

ϵ =0.5: Minimum required amount (in 100 g) of a category g in meal k.

notRecommended(i,k): Binary parameter,equal to 1 if food i is not recommended for a meal (example dinner $k = 3$), otherwise 0.

λ : Penalty for violating the "not recommended for dinner" constraint.

Constraints

1 Nutrient Constraints

Ensure the total nutrient intake meets or exceeds daily requirements:

$$\sum_{i=1}^{N} \sum_{k=1}^{3} A_{ij} x_{i,k} \geq B_j,\ \forall j = 1,\ldots,M$$

2 Total Quantity Limit

The total quantity of all food items per day must not exceed maxTotalQuantity=$60(6\ \mathrm{kg})$:

Latifa.dekhici@univ-usto.dz

Operational Research

$$\sum_{i=1}^{N}\sum_{k=1}^{3} x_{i,k} \leq 60$$

3 Per Item Quantity Limit

The quantity of any food item i must not exceed its maximum allowed quantity q_{max} as an example no more than 3 (300 g) of banana.

$$x_{i,k} \leq \mathrm{qmax}_i,\ \forall i \in I, \forall k \in K.$$

4 Meal Composition (Categories)

Ensure each meal includes at least one item from each required category.This constraint ensures that for each meal k and each category g, the total quantity of food items belonging to that category is at least ϵ (e.g., 10 g).

:

$$\sum_{i=1}^{N} \mathrm{cat}\,(i,g)\cdot x_{i,k} \geq \epsilon,\ \forall g \in G, \forall k \in K.$$

5 Non-recommended Items at Dinner

Introduce a penalty with coefficient λ including non-recommended items in dinner $(k = 3)$ base on this soft constraint

$$\sum_{i=1}^{N} notRecommanded_{i3} x_{i,3} = 0$$

Objective Function
Minimize the total cost, including penalties for violating soft constraints:

$$\text{Minimize: } Z - \sum_{i=1}^{N}\sum_{k=1}^{3} c_i x_{ik} + \lambda \sum_{i=1}^{N} notRecommanded_{i3} x_{i,3}$$

Objective:
You are tasked with implementing a local search algorithm to solve the diet optimization problem defined below. The goal is to minimize the total cost while meeting nutritional requirements and ensuring balanced meal composition.

This project will be completed in many stages:

1. Add to the model a maximum allowed nutrient Bmax_j.
2. Implement a function that reads all data from an excel, cvs or text file(N, M, Aij,Bj,Ci…).
3. Implement a function that generate a random feasible solution without abusing hard constraint.
4. Implement the cost function that evaluates the cost of any solution X.
5. Implement the objective function that evaluates any solution X using the cost function plus the penalties to abuse soft constraint.
6. Implement a neighborhood function that create a neighbor from a solution X.

Latifa.dekhici@univ-usto.dz

Operational Research

7. Implement a function that corrects any solution X to be in the definition domain.
8. Implement a function that evaluate the comparison function F(x)=f(x)+α constraintsAbuse(X)
9. Implement a local search algorithm that finds an improved solution to the problem, starting from an initial feasible solution and a stop criterion in its parameters and using neighborhood function and the comparison function F and provides as results the best solution with the best cost.
10. Launch data import using the annex, and the local search algorithm from the main function
11. Display the best solution, its cost, and the evaluation of its nutrients.
12. Implement a function that generates a population of size=10 or 20 initial solutions using function created in 3.
13. Implement interior search algorithm or particle swarm optimization with size and stop criterion and a population of size solutions as parameters and provide the best solution and its cost.
14. Launch interior search algorithm from the main function using
15. Optional: Choose another recent metaheuristic(squid game optimizer, mother optimization algorithm, Waterwheel Plant Algorithm…) and compare it with the existing ones using the same initial population. You can improve the algorithms by adding the running time to get the best solution for each one. You can improve the program by adding many simulations in the main function. You can improve the program by exporting the results to the results to an excel or cvs file.