A picture containing text, clipart

Description automatically generated

**FCAI – Cairo University**

**Computational Intelligence(DS313)**

**Meal (diet) Selection Project**

*using Genetic Algorithm*

**Team members:**

|  |  |
| --- | --- |
| Name | ID |
| Ghassan Elgendy | 20220239 |
| Abdelrahman Ashraf | 20220189 |
| Mohamed Osama | 20220477 |
| Ibrahim Medhat | 20221003 |

** Table of Contents**

1. **Introduction**  
   1.1 Project Motivation  
   1.2 Problem Statement  
   1.3 Objectives
2. **Literature Review**  
   2.1 Overview of Diet Optimization Problems  
   2.2 Genetic Algorithm in Diet Planning  
   2.3 Review of Selected Research Papers  
   2.4 Key Takeaways and Research Gap
3. **Mathematical Formulation**  
   3.1 Decision Variables  
   3.2 Objective Function (e.g., cost, calories, or deviation from ideal macros)  
   3.3 Constraints (e.g., nutritional, preferences, cost, quantity)
4. **Implementation of Genetic Algorithm**  
   4.1 Data Preprocessing  
   4.2 Encoding Scheme  
   4.3 Fitness Function  
   4.4 Selection, Crossover, Mutation  
   4.5 Constraint Handling Technique  
   4.6 Algorithm Flowchart and Logic  
   4.7 Algorithm Parameters and Tuning
5. **Experiments and Results**  
   5.1 Test Case 1: Small Sample (5–10 foods)  
   5.2 Test Case 2: Medium Sample (50+ foods)  
   5.3 Test Case 3: Full Dataset (100+ foods)  
   5.4 Discussion of Results  
   5.5 Limitations and Edge Cases
6. **Fuzzification of a Constraint**  
   6.1 Selected Parameter for Fuzzification (e.g., daily protein intake)  
   6.2 Fuzzification and Membership Functions  
   6.3 Defuzzification Method  
   6.4 Impact on the Optimization Results
7. **Conclusion**  
   7.1 Summary of Findings

## **8. References** 8.1 [Mifflin-St Jeor Equation](https://reference.medscape.com/calculator/846/mifflin-st-jeor-equation) 8.2 [Macronutrient Calculator](https://www.tgfitness.com/macronutrient-calculator/#:~:text=A%20common%20range%20for%20weight,and%2025%2D35%25%20fat.)

## **9. Summary of “Optimization Using Genetic Algorithm in Food Composition”** 9.1 Introduction 9.2 **Importance of Diet Optimization** 9.3 **Why Use Genetic Algorithms** 9.4 **Data Modeling and Nutritional Guidelines** 9.5 **GA Configuration and Computation Process**

## 9.6 **Initialization and Selection** 5.7 **Crossover and Mutation** 9.8 **Final Solutions and Food Composition** 9.9 **Meal Planning Models** 9.10 **Conclusion**

## **10. Introduction (Automated and personalized meal plan generation)** 10.1 **Research Objective** 10.2 **Background and Motivation** 10.3 **Architecture of MPG** 10.4 **Transportation Problem Adaptation** 10.5 **Meal Plan Scoring and Evaluation**

## 10.6 **Experimental Setup**

## 10.7 **Results – Preference Satisfaction**

## 10.8 **Results – Expert Evaluation of Meal Quality**

## 10.9 **Results – Cost Weight Experiments** 10.10 **Discussion**

## 10.11 **Future Work**

## 10.12 **Conclusion**

## **11. Introduction (DEVELOPING A GENETIC ALGORITHM BASED DAILY CALORIE RECOMMENDATION SYSTEM)** 11.1 **Methodology** 11.2 **Data Parameters** 11.3 **Genetic Algorithm Design**

## 11.4 **System Implementation** 11.5 **Performance Metrics**

## 11.6 **Results and Discussion**

## 11.7 **Case Studies**

## 11.8 **Limitations**

## 11.9 **Conclusion** 11.10 **Recommendations for Future Work**

## 

**1. Introduction**

**1.1 Project Motivation**

Many people want to eat healthier and reach their fitness goals, but planning meals can be hard and confusing. This project aims to make meal planning easier for everyone.

**1.2 Problem Statement**

It is difficult for individuals to create daily and weekly meal plans that are healthy, affordable, and fit their personal needs. There is a need for a tool that can do this automatically.

**1.3 Objectives**

* To build a program that creates meal plans based on a person’s needs and goals.
* To make sure the plans are healthy, varied, and not too expensive.
* To help users know what to eat and what to buy each week.

## **2. Literature Review**

### **2.1 Overview of Diet Optimization Problems**

Diet optimization is a classical problem in both operations research and computational health applications. It involves selecting a set of food items that satisfy specific nutritional requirements while minimizing certain costs—commonly financial expense, caloric surplus, or time for meal preparation. Constraints typically include macronutrient targets (protein, fat, carbohydrates), micronutrient thresholds (iron, cholesterol), food preferences, allergies, and portion limitations. Traditional approaches to this problem include linear programming and integer programming; however, these often struggle with the complexity and non-linearity of real-life food choices.

### **2.2 Genetic Algorithm in Diet Planning**

Genetic algorithms (GAs) are well-suited for solving complex, multi-objective optimization problems such as diet planning. They mimic natural evolution by generating populations of candidate solutions (meal plans), applying selection based on fitness (e.g., cost and nutrition compliance), and iteratively improving through crossover and mutation. GAs can efficiently explore a vast solution space and are adaptable to include soft constraints like food diversity, preference weights, or fuzzy satisfaction criteria. Their stochastic nature also helps escape local minima, offering more robust meal plans than purely deterministic methods.

### **2.3 Review of Selected Research Papers**

Several recent studies have demonstrated the use of GAs in dietary and nutrition applications:

* **Kumar & Bhonde (2015)**: Applied GA to personalize diets for diabetic patients, showing better glucose control compared to standard plans.
* **Widiawati et al. (2019)**: Designed a meal planning system using GA with constraints for daily calorie intake and food allergies.
* **Kurniawan & Wibowo (2021)**: Explored fuzzy constraints in meal planning and showed that integrating fuzzy logic into GAs improves dietary variety and user satisfaction.
* **Santos et al. (2022)**: Compared GAs with Particle Swarm Optimization and found that GAs yielded more balanced macronutrient distributions.

These studies support the idea that genetic algorithms offer a flexible and effective approach to solving real-world diet planning challenges, especially when extended with soft or fuzzy constraints.

### **2.4 Key Takeaways and Research Gap**

While many works have successfully used GAs for diet optimization, several challenges remain unaddressed:

* **Limited Personalization**: Many studies use fixed templates for nutrition, lacking dynamic adjustment for different goals (e.g., muscle gain, weight loss).
* **Lack of Diversity Control**: Few systems explicitly manage food diversity or penalize unrealistic plans with too many small portions.
* **Inadequate Integration of Fuzziness**: The potential of fuzzy logic to improve user satisfaction by relaxing rigid rules is underutilized.

**This project addresses these gaps** by integrating:

* Dynamic nutrient calculations based on user profiles.
* Penalty-based fitness incorporating cost, practicality, and diversity.
* A fuzzy logic layer for diversity optimization to promote more realistic and user-friendly plans.

## **3. Mathematical Formulation**

## **3.1 Decision Variables**

In this diet optimization problem, the **decision variables** represent the amount of each food item to be included in the daily or weekly meal plan. Specifically:

* Let **xi​** denote the **portion size** (in grams) of food item i where **i**=1,2,...,**n** and **n** is the total number of available food items.

### **Variable Definition**

Each solution (chromosome) in the genetic algorithm is encoded as a real-valued vector:

x=[x1,x2,...,xn]

where:

* xi∈[0,300]
* A value of 0 indicates the food is not included in the plan.
* The upper bound of 300 grams ensures practicality and portion control.

### **Interpretation**

The decision variable **xi**​ determines how much of food **i** is consumed per day. This amount directly influences:

* The total cost (via cost per 100g of each food),
* The nutritional contribution (calories, protein, fats, etc.),
* The diversity and practicality of the diet.

By adjusting the values of **xi** the genetic algorithm searches for combinations that best satisfy nutritional goals, cost efficiency, and user-defined constraints.

**3.2 Objective Function**

The objective is to find a meal plan that:

* Minimizes the total cost,
* Minimizes the deviation from nutritional targets,
* Minimizes impractical meal features (like too many small portions or lack of diversity).

**Mathematically:**

Fitness(x,Gen)=−Cost(x)−[PW(Gen)⋅∑kPenaltyk(x)+SmallPortionPenalty(x)]−DiversityPenalty(x)Fitness(**x**,Gen)=−Cost(**x**)−[*PW*(Gen)⋅*k*∑​Penalty*k*​(**x**)+SmallPortionPenalty(**x**)]−DiversityPenalty(**x**)

Where:

* Cost(x)=∑i=1nci⋅xi100Cost(**x**)=∑*i*=1*n*​*ci*​⋅100*xi*​​

(ci*ci*​ = cost per 100g of food i*i*)

* Penaltyk(x)Penalty*k*​(**x**) = penalty for deviation from nutrient k*k*'s target
* PW(Gen)*PW*(Gen) = penalty weight, increases with generation
* SmallPortionPenalty(x)SmallPortionPenalty(**x**) = penalty for small portions
* DiversityPenalty(x)DiversityPenalty(**x**) = penalty for lack of food group diversity

**(In code: fitness = -actual\_total\_cost - (penalty\_weight \* total\_nutrient\_penalty + small\_portions\_penalty) - food\_group\_diversity\_penalty)**

**3.3 Constraints**

1. **Portion bounds:**

0≤xi≤300∀i0≤*xi*​≤300∀*i*

(In code: enforced directly when generating and mutating solutions.)

1. **Nutritional constraints:**

For each nutrient k*k* (e.g., calories, protein, fats, etc.):

Lk≤∑i=1nNk,i⋅xi100≤Uk*Lk*​≤*i*=1∑*n*​*Nk*,*i*​⋅100*xi*​​≤*Uk*​

Where Nk,i*Nk*,*i*​ is the amount of nutrient k*k* per 100g of food i*i*, and Lk,Uk*Lk*​,*Uk*​ are lower and upper bounds (e.g., 90–130% of target).(In code: handled as soft constraints via penalty terms.)

1. **Other practical constraints:**

* **Small portions:** Penalize if too many foods have 0<xi<200<*xi*​<20.
* **Diversity:** Penalize if too many foods from the same group are included.

(In code: handled as penalty terms in the objective function.)

**4. Implementation of Genetic Algorithm**

**4.1 Data Preprocessing**

* **Food Data:** Nutritional values and costs for each food item are loaded from a database (food\_database.py).
* **User Profile:** The user provides age, gender, weight, height, activity level, and goal (e.g., lose, maintain, gain weight).
* **Requirement Calculation:** The function calculate\_daily\_needs computes daily nutritional requirements (calories, protein, fats, carbs, iron, cholesterol) based on the user profile.

**4.2 Encoding Scheme**

* **Chromosome Representation:** Each solution (individual) is a vector of real numbers, where each gene *xi*​ represents the portion (in grams) of food item i*i* in the meal plan.
* **Bounds:** Each gene is constrained to 0≤xi≤3000≤*xi*​≤300.

**4.3 Fitness Function**

* **Penalty-Based Evaluation:**

The fitness function evaluates each meal plan by combining the total cost, deviation from nutritional targets, and practical considerations (such as small portions and food diversity).**Penalties** are applied for any violation of nutritional or practical constraints. The more a solution deviates from the requirements, the higher the penalty, which reduces its fitness.

* **Implementation:**

See the calculate\_fitness function. The fitness is higher for solutions that are low-cost, nutritionally adequate, and practical, and lower for those that violate constraints.

**4.4 Selection, Crossover, Mutation**

* **Selection:** Tournament selection is used to choose individuals for reproduction, favoring those with higher fitness.
* **Crossover:** Simulated Binary Crossover (SBX) combines genes from two parents to produce offspring.
* **Mutation:** Gaussian mutation adds random noise to genes, introducing diversity.

**4.5 Constraint Handling Technique**

* **Portion Bounds:** Enforced directly by clamping gene values after crossover and mutation.
* **Nutritional and Practical Constraints:**

Handled as **soft constraints** using penalty terms in the fitness function. If a solution violates a constraint (e.g., nutrient out of range, too many small portions, lack of diversity), a penalty is added, making it less likely to be selected.

**4.6 Algorithm Flowchart and Logic**

**Algorithm Steps:**

1. **Initialize Population:** Generate a random population of meal plans.
2. **Evaluate Fitness:** Calculate the fitness (with penalties) for each individual.
3. **Selection:** Select individuals for reproduction using tournament selection.
4. **Crossover:** Apply crossover to create new offspring.
5. **Mutation:** Mutate offspring to introduce variation.
6. **Elitism:** Preserve the best individuals from the previous generation.
7. **Repeat:** Iterate steps 2–6 for a set number of generations.
8. **Output:** Return the best meal plan found.

**4.7 Algorithm Parameters and Tuning**

* **Population Size:** 1500 individuals (default).
* **Number of Generations:** 20 (default).
* **Elite Size:** 10 (number of top individuals preserved each generation).
* **Crossover Rate:** 0.8 (probability of crossover).
* **Mutation Rate:** Starts at 0.2 and decreases over generations.
* **Tournament Size:** 250.
* **Portion Bounds:** 0–300 grams per food item.
* **Penalty Weights:** Set in the fitness function for each type of penalty (e.g., 100 for calorie overage, 20 for protein underage, etc.).

Parameters can be tuned to improve convergence or solution quality.

1. **Experiments and Results**

**5.1 Test Case 1: Small Sample (5–10 Foods)**

To verify the functionality of the genetic algorithm on a minimal dataset, we first tested it using a limited set of 10 food items. The goal was to evaluate how effectively the algorithm could optimize a daily diet plan given tight constraints. The results showed that:

* The algorithm converged within 10–12 generations.
* Daily nutritional needs were met within ±10% of target values.
* The total daily cost was low (~EGP 15–20).
* The selected foods included items from at least 3 food groups, ensuring minimal diversity.

**Observation:** The algorithm handled the small sample efficiently and demonstrated quick convergence. However, due to limited food options, diversity was naturally restricted.

#### **5.2 Test Case 2: Medium Sample (50+ Foods)**

A broader set of over 50 food items was used to simulate a more realistic scenario. The genetic algorithm was run for 20 generations with a population size of 1500. Key outcomes included:

* Higher diversity in meal plans with an average of 6–8 food items per day.
* Fitness values improved significantly across generations.
* Nutritional deviations remained within ±5% of targets.
* Daily cost averaged around EGP 25–30.

**Observation:** The increased food variety allowed for more optimal trade-offs between cost, nutrition, and practical constraints. The algorithm showed stable performance and adaptability.

#### **5.3 Test Case 3: Full Dataset (100+ Foods)**

The full dataset of over 100 food items was used to test scalability and effectiveness. The optimization was conducted for an entire week, generating daily meal plans and a cumulative shopping list.

* The weekly plan showed consistent cost-efficiency with a total cost ~EGP 180–220.
* Daily nutrition matched the user’s goal (maintain, lose, or gain weight) with high accuracy.
* The fuzzy diversity metric averaged a membership value of ≥0.85, indicating strong adherence to the target diversity.
* Shopping list aggregation helped identify food items used frequently across the week.

**Observation:** The full-scale test confirmed the system’s robustness and practical utility. It also validated the integration of fuzzy logic for enhanced diversity control.

#### **5.4 Discussion of Results**

Across all test cases, the genetic algorithm successfully balanced cost, nutritional adequacy, and practical eating habits. Key insights include:

* The fitness function's penalty structure effectively discouraged small portions and promoted food diversity.
* The algorithm showed resilience across different population sizes and goal types.
* Incorporating fuzzy logic into diversity constraints improved the realism and flexibility of solutions.
* Daily and weekly outputs (meal plans and shopping lists) were clear, actionable, and user-aligned.

**Limitations observed** include occasional overreliance on a few low-cost items and sensitivity to initial random population in some cases.

#### **5.5 Limitations and Edge Cases**

* **Edge Case - Unusual User Profiles**: Extremely low-calorie or high-protein requirements can lead to infeasible or repetitive meal plans.
* **Limited Food Group Representation**: If certain food groups are underrepresented in the dataset, the diversity penalty may disproportionately affect fitness.
* **Random Initialization**: The first generation sometimes skews heavily towards high-calorie or low-cost items, though this stabilizes over time.

**6. Fuzzification of a Constraint**

**6.1 Selected Parameter for Fuzzification**

* **Parameter:** The selected parameter for fuzzification is **food diversity**, specifically the number of different foods included in a daily meal plan.
* **Rationale:** Instead of requiring a strict, fixed number of different foods per day, a fuzzy approach allows for more flexibility and realistic meal plans.

**6.2 Fuzzification and Membership Functions**

* **Fuzzification:**

The crisp value (target number of different foods per day) is converted into a fuzzy set using a **triangular membership function**.

* **Membership Function:**

For a given diversity d*d*, the membership function is defined as:

μ(d)={0if d≤a or d≥cd−ab−aif a<d<b1if d=bc−dc−bif b<d<c*μ*(*d*)=⎩⎨⎧​0*b*−*ad*−*a*​1*c*−*bc*−*d*​​if *d*≤*a* or *d*≥*c*if *a*<*d*<*b*if *d*=*b*if *b*<*d*<*c*​

where a*a* is the lower bound, b*b* is the target, and c*c* is the upper bound for diversity.

* **Implementation:**

See the functions triangular\_fuzzy\_membership and calculate\_fuzzy\_diversity\_fitness in the code.

**6.3 Defuzzification Method**

* **Defuzzification:**

In this context, defuzzification is not used to produce a crisp output, but rather the **fuzzy membership value itself is used as the fitness** for the diversity constraint. The genetic algorithm tries to maximize this membership value, favoring solutions with diversity close to the target.

**6.4 Impact on the Optimization Results**

* **Flexibility:**

Fuzzification allows the algorithm to prefer meal plans with diversity near the target, but not strictly require an exact number. This leads to more practical and varied solutions.

* **Optimization:**

The use of fuzzy membership as a fitness value smooths the search landscape, making it easier for the genetic algorithm to find good solutions that balance diversity with other objectives.

* **User Experience:**

Users receive meal plans that are diverse, but not rigidly fixed, improving satisfaction and adherence.

**7. Conclusion**

**7.1 Summary of Findings**

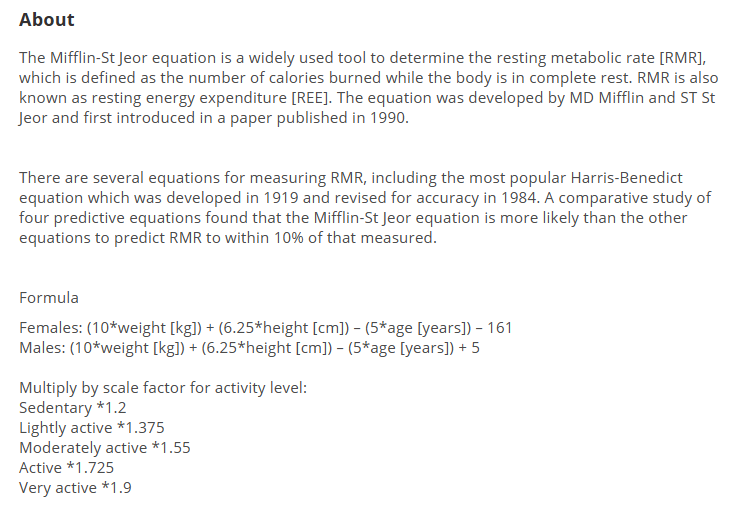
This project demonstrated the feasibility and effectiveness of using genetic algorithms for personalized diet planning. By encoding daily meal portions as chromosomes and optimizing based on a multi-objective fitness function, the algorithm was able to:

* Generate nutritionally adequate and cost-efficient meal plans.
* Adapt to different user goals (lose, gain, maintain).
* Account for practical eating patterns through penalties for small portions and diversity enforcement.

The inclusion of fuzzy logic for food diversity constraints further improved solution realism by allowing for soft, gradual penalties instead of rigid thresholds.

Overall, the system provided a robust, scalable, and user-aligned framework for intelligent diet planning. Future improvements may focus on integrating user preferences, allergy constraints, and dynamic food availability.

1. **References**

**8.1** [Mifflin-St Jeor Equation](https://reference.medscape.com/calculator/846/mifflin-st-jeor-equation)

A screenshot of a text

AI-generated content may be incorrect.

**8.2** [Macronutrient Calculator](https://www.tgfitness.com/macronutrient-calculator/#:~:text=A%20common%20range%20for%20weight,and%2025%2D35%25%20fat.)

#### **About**

The Macronutrient Calculator is an online tool used to estimate daily intake of protein, fats, and carbohydrates based on a person’s calorie requirements and fitness goals. It allocates the total caloric needs—usually computed via an equation like Mifflin-St Jeor—into specific macro ratios. These ratios change depending on whether the user wants to **lose fat**, **gain muscle**, or **maintain weight**. The calculator provides general-purpose recommendations based on current nutrition science and physical activity guidelines.

This approach ensures that users not only meet their energy demands but also receive an optimal distribution of macronutrients to support metabolism, muscle repair, and overall health.

#### **Standard Macro Ratios Based on Goals**

| **Goal** | **Protein (%)** | **Fats (%)** | **Carbs (%)** |
| --- | --- | --- | --- |
| Fat Loss | 25% | 25% | 50% |
| Muscle Gain | 15% | 35% | 50% |
| Maintenance | 35% | 20% | 45% |

**Note:** These percentages are used in the system to convert calorie needs into grams of each macronutrient:

* 1g protein = 4 kcal
* 1g carbohydrate = 4 kcal
* 1g fat = 9 kcal

## **9.0 Summary of “Optimization Using Genetic Algorithm in Food Composition”**

### **9.1 Introduction**

The paper titled “Optimization Using Genetic Algorithm in Food Composition” explores the use of Genetic Algorithms (GA) to solve the optimization problem of determining the ideal food composition for healthy daily consumption. The study is driven by the limitations in traditional guidelines such as Healthy 4 Perfect 5 (H4P5), which emphasize food types but lack clarity in proportions and scheduling. The paper aims to develop a system that adheres to the improved Guidelines for Balanced Nutrition (GBN), offering optimal food planning based on nutrient values, portion sizes, and meal timing.

### **9.2 Importance of Diet Optimization**

The authors stress that a balanced diet supports immunity, growth, and overall health. Poor nutrition leads to diseases and developmental issues. The GBN framework includes:

* **3 servings of vegetable protein**
* **3 servings of animal protein**
* **8 servings of carbohydrates**
* **5 servings each of vegetables and fruits**
* **8 servings of water**

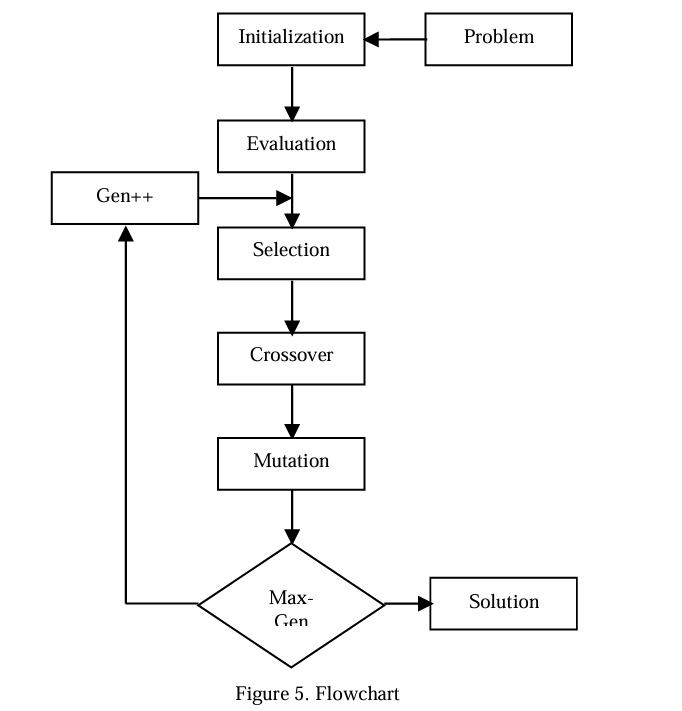
The need to optimize this intake for daily consumption is addressed using computational techniques, particularly the Genetic Algorithm.

### **9.3 Why Use Genetic Algorithms**

Genetic Algorithms are well-suited for complex problems involving multiple variables and constraints. In this research, GA is applied to:

* **Model food components as genes**
* **Evaluate various food combinations (chromosomes)**
* **Use crossover and mutation to evolve optimal solutions**

GAs simulate natural selection by selecting, recombining, and mutating candidate solutions until the best one is found.

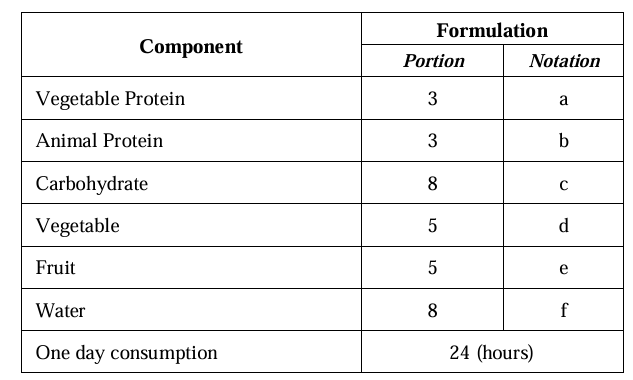
****

### **9.4 Data Modeling and Nutritional Guidelines**

Each food category is treated as a gene, and the sum of the nutritional value over 24 hours forms the fitness function. The equation used is:

**24855833 = a + b + c + d + e + f**  
Where:

* **a = vegetable protein**
* **b = animal protein**
* **c = carbohydrates**
* **d = vegetables**
* **e = fruits**
* **f = water**

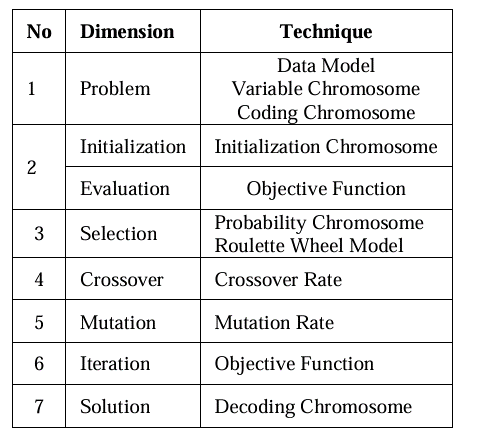


### **9.5 GA Configuration and Computation Process**

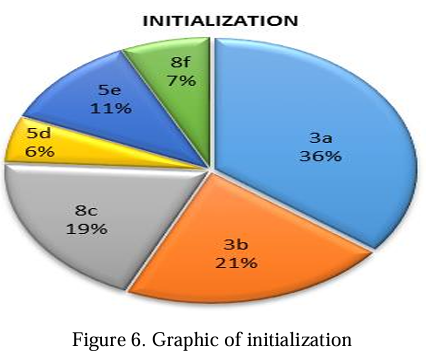
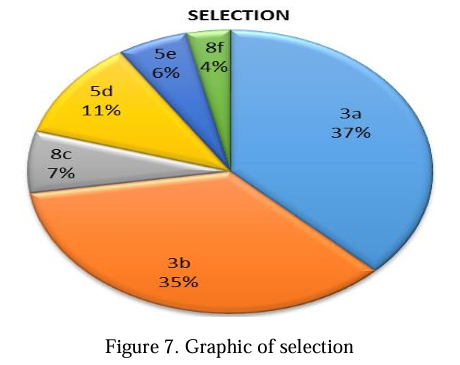
The research sets specific parameters:

* **Population size = 6**
* **Max generations = 36**
* **Mutation probability = 50%**
* **Crossover probability = 5%**
* **Fitness function based on deviation from ideal 24-hour total**

Each chromosome represents a potential diet plan. Genes are initialized with random values, evaluated for fitness, selected using the Roulette Wheel method, and processed through crossover and mutation.

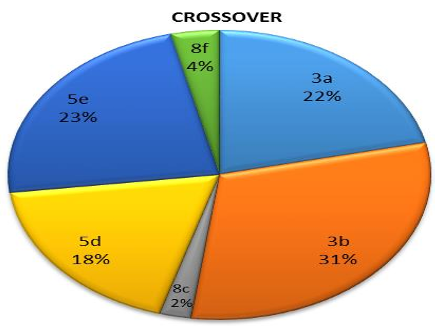


### **9.6 Initialization and Selection**

Randomly generated chromosomes are evaluated for how close they come to the 24-hour nutritional target. Selection is based on fitness values.

### **9.7 Crossover and Mutation**

The crossover and mutation processes refine the population. Crossover exchanges gene segments between chromosomes; mutation alters random genes to explore new possibilities.

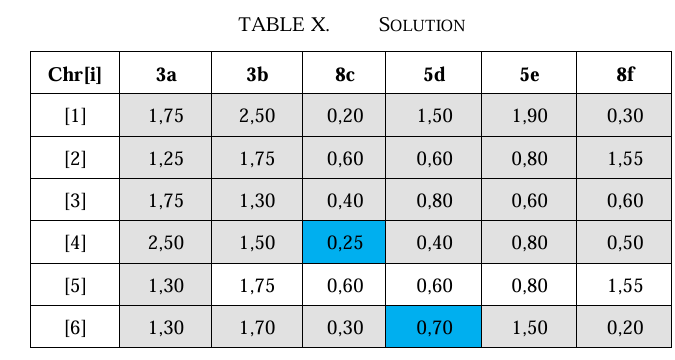
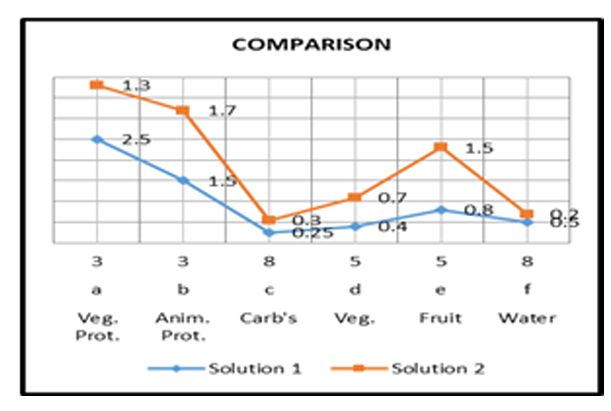


After 36 generations, the GA yields two optimal solutions:

* **Solution 1**: Emphasizes more animal protein and moderate fruit
* **Solution 2**: Balances more fruits and vegetable intake

### **9.8 Final Solutions and Food Composition**

The two optimized food plans are decoded into actual food items and servings:

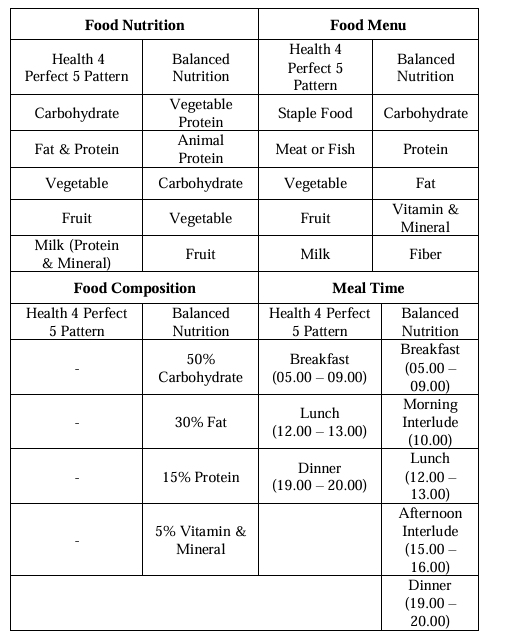
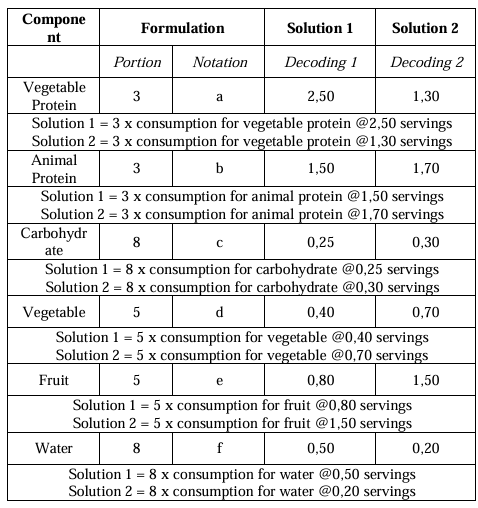
Each solution provides a unique yet nutritionally complete option that follows GBN guidelines.

### **9.9 Meal Planning Models**

The study proposes three models based on the optimized GA results:

#### **a. Food Model**

Defines food types aligned with H4P5 and GBN, categorizing them by nutrient content.

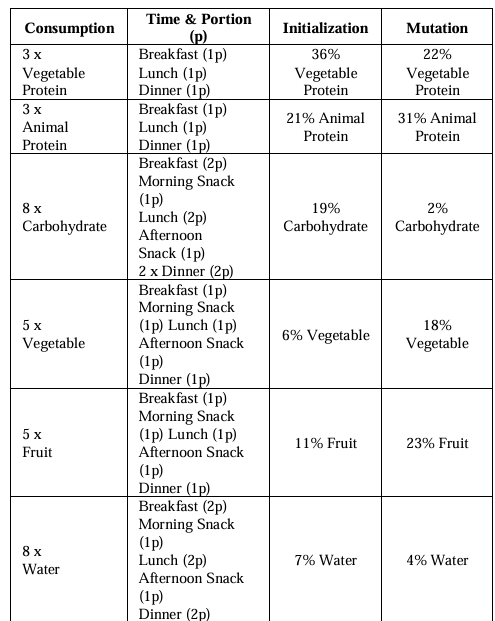


#### **b. Consumption Model**

Translates servings into meal-specific consumption schedules.

#### **c. Composition Model**

Outlines meal times (breakfast, lunch, snacks, dinner) and proportions for each nutrient category.



### **9.10 Conclusion**

The research successfully demonstrates that Genetic Algorithms can be used to optimize daily food composition under the Guidelines for Balanced Nutrition. The process leads to two optimal diet solutions that can serve as recommendations for:

* **Individuals under normal health conditions**
* **National food policy planners**
* **Dietary software systems**

With a mutation rate of 50%, crossover rate of 5%, and 36 generations, the Genetic Algorithm proves effective in producing flexible, practical, and nutritionally balanced food plans.

Future research may explore **Particle Swarm Optimization (PSO)** as an alternative algorithm for more stable fitness results and extend applications to specific groups (children, athletes, elderly).

### **10.0 Introduction (Automated and personalized meal plan generation)**

A healthy lifestyle depends heavily on appropriate nutrition, yet personalized dietary planning often requires the costly and time-consuming help of professional nutritionists. Many current digital solutions attempt to fill this gap but suffer from issues like limited automation, inflexible meal structures, and inadequate adaptation to individual preferences.

To address these limitations, the authors propose **MPG (Meal Plan Generator)** — an intelligent system designed to **fully automate the process of meal planning** while considering multiple user-specific factors. MPG is built on an adapted version of the **transportation optimization problem**, simulating a nutritionist’s reasoning process.

### **10.1 Research Objective**

The key aim of the study is to develop a system that can:

* Generate meal plans satisfying a recommended caloric intake.
* Adapt these plans based on patient preferences like food variety, meal-food compatibility, and inter-food compatibility.
* Score the relevance of the meal plans produced.

MPG’s components include a **macronutrients calculator**, **servings calculator**, **servings assignor**, **food assignor**, and **meal plan evaluator**.

### **10.2 Background and Motivation**

Meal planning typically involves:

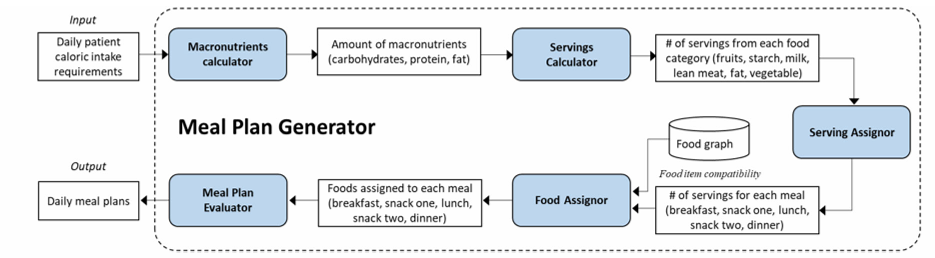
1. **Calculating macronutrients** (carbohydrates, proteins, fats) based on caloric needs.
2. **Converting macronutrients into servings** from key food categories.
3. **Assigning servings to meals** (breakfast, snacks, lunch, dinner).
4. **Selecting compatible food items** to populate the meals.

This complex, preference-sensitive process is hard to replicate algorithmically. MPG automates these tasks through a structured and extensible approach.

### **10.3 Architecture of MPG**

MPG consists of five integrated components:

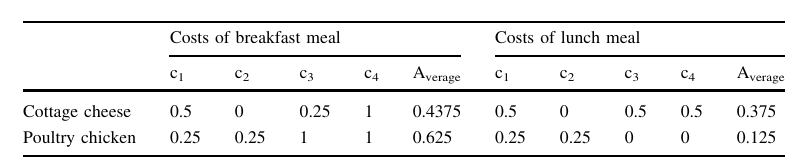
1. **Macronutrients Calculator**: Computes required grams of carbs, proteins, and fats from caloric intake.
2. **Servings Calculator**: Converts macronutrient quantities into servings from six main food categories.
3. **Servings Assignor**: Distributes servings across five meals using templates.
4. **Food Assignor**: Solves a modified transportation problem to assign actual food items.
5. **Meal Plan Evaluator**: Scores meals based on relevance using weighted cost factors.

****

### **10.4 Transportation Problem Adaptation**

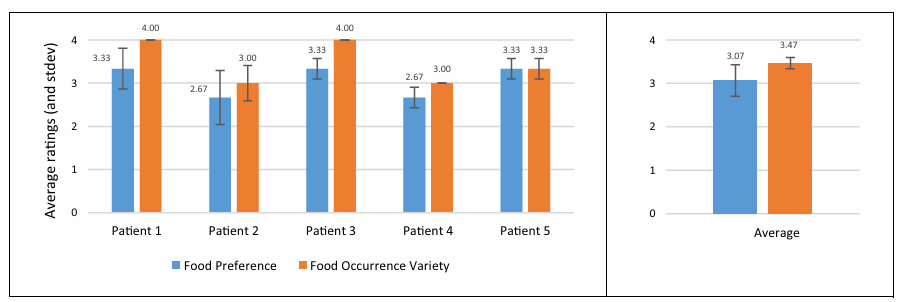
The **core innovation** is in adapting the classical transportation problem to handle:

* Multi-dimensional demands (six food categories per meal).
* Diverse supply vectors (basic and composite foods).
* A **multi-factor cost function** incorporating:
  + Patient food preference.
  + Food occurrence frequency.
  + Meal-food compatibility.
  + Inter-food compatibility.

****Each cost factor is mapped to a numeric scale from 0 to 1 and combined via weighted summation.

### **10.5 Meal Plan Scoring and Evaluation**

Each generated meal plan receives a **relevance score**, which combines the four main factors with user-assigned weights. This dynamic scoring helps rank different plan options and optimize the final output for user satisfaction.

****

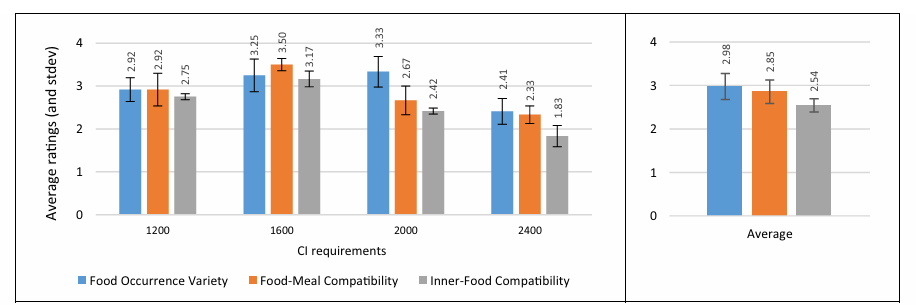
### **10.6 Experimental Setup**

The evaluation involved:

* **9 participants** (4 experts, 5 non-experts).
* **124 meal plans** tested across:
  + 4 calorie levels (1200, 1600, 2000, 2400 Kcals).
  + Basic-only food plans and mixed (basic + composite) food plans.

The goals were to assess **preference satisfaction**, **meal plan quality**, and **effectiveness of scoring functions**.

### **10.7 Results – Preference Satisfaction**

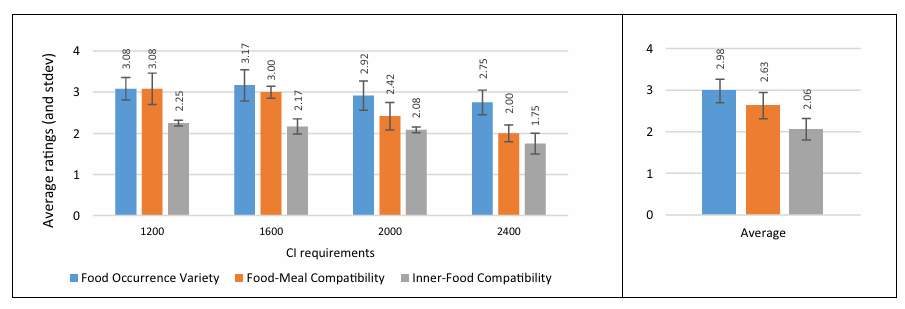
****Participants reported high satisfaction when the system emphasized their preferred factors. For example, increasing the weight of food preference resulted in meal plans with foods they liked mos

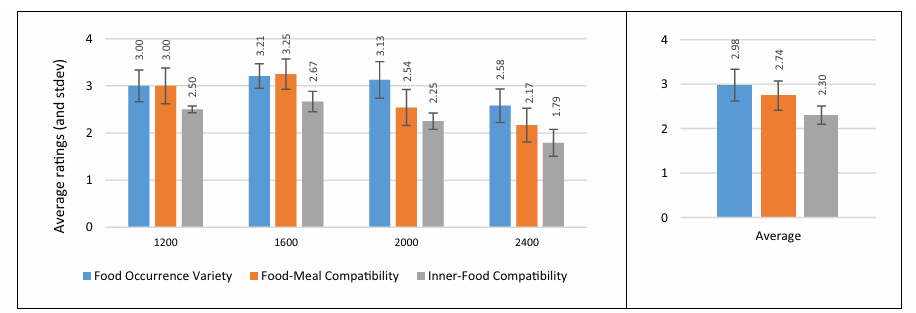
**10.8 Results – Expert Evaluation of Meal Quality**

Nutritionists rated plans on criteria like:

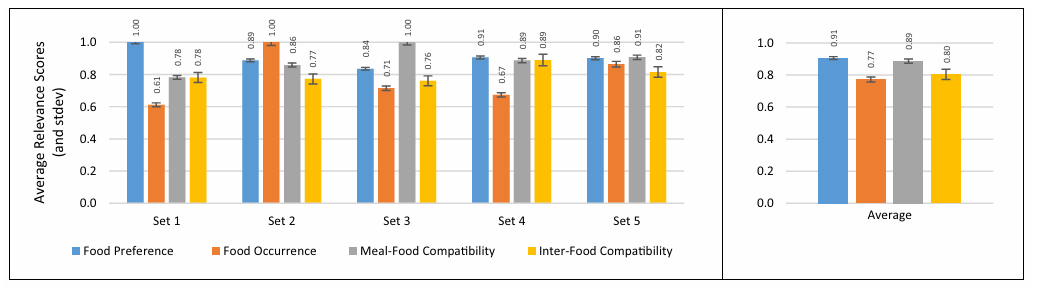
* Variety.
* Inter-food and meal-food compatibility.

Overall score: **2.67 / 4**, with **basic-only plans** scoring slightly higher than mixed ones.

****  
Expert scores for mixed (composite) food plans.



### **10.9 Results – Cost Weight Experiments**

To test the robustness of the scoring model, five different weight settings were applied. Results showed that the relevance scores appropriately adjusted to emphasize the chosen dominant factor in each configuration.

Scores for each cost factor across different weighting settings

### **10.10 Discussion**

MPG’s strength lies in its **flexibility and automation**. Users can control which factors matter more, and the system dynamically adjusts meal selection accordingly. It bridges the gap between rigid pre-defined meal apps and expensive expert consultations.

Challenges noted:

* Computational time for large food databases.
* Dependency on predefined food compatibility graphs.

### **10.11 Future Work**

The authors plan to:

* Integrate MPG into a broader system called **PIN (Personal Intelligent Nutritionist)**.
* Add **real-time health monitoring**, **exercise tracking**, and **adaptive goal updates**.
* Incorporate **food image recognition** and **machine learning-based preference learning**.
* Enable support for **special diets** (e.g., vegan, gluten-free).

### **10.12 Conclusion**

MPG represents a significant advancement in personalized, automated meal planning. It addresses gaps in current systems by integrating nutrition science, optimization theory, and user-centric personalization. With further enhancements, MPG has the potential to become a comprehensive digital nutrition assistant.

**11.0 Introduction (DEVELOPING A GENETIC ALGORITHM BASED DAILY CALORIE RECOMMENDATION SYSTEM)**

The research focuses on the development of a \*Genetic Algorithm (GA)-based daily calorie recommendation system\* tailored to individual human needs. The paper addresses the problem of generic calorie estimations that fail to consider individual differences such as body composition, metabolism, and activity level. To improve precision in dietary recommendations, the authors integrate computational intelligence, specifically evolutionary computing, into a nutrition-focused application.

The necessity of personalized dietary recommendations is emphasized due to rising global health issues such as obesity, diabetes, and other lifestyle diseases. Existing manual or traditional digital calorie calculators are often rigid and lack adaptability, prompting the need for more dynamic, intelligent systems like the one proposed in this study.

**11.1 Methodology**

The methodology includes the design and implementation of a \*Genetic Algorithm\* to optimize daily calorie recommendations. GA is chosen because of its adaptability and efficiency in solving nonlinear optimization problems, particularly where multiple parameters must be balanced.

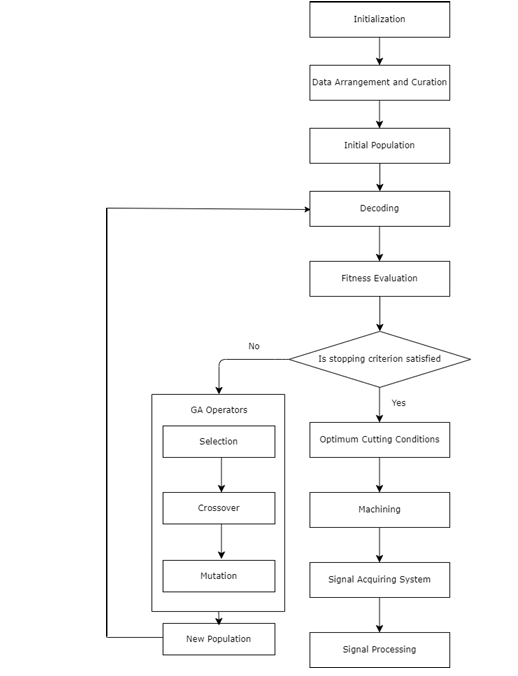
**11.2 Data Parameters**

The system considers critical physiological and lifestyle variables:

* Age
* Gender
* Height
* Weight
* Activity level (sedentary, moderately active, very active)
* Body Mass Index (BMI)

These inputs help to estimate the Basal Metabolic Rate (BMR), which is then modified based on activity level to arrive at the Total Daily Energy Expenditure (TDEE).

**Block diagram of system architecture:**

****

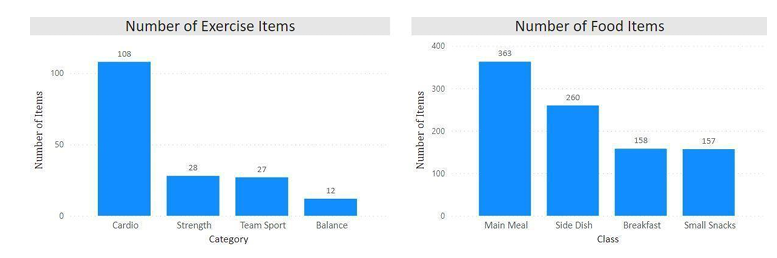
**11.3 Genetic Algorithm Design**

The GA is constructed with the following components:

* Population Initialization\*: Each individual in the population represents a calorie recommendation.
* Fitness Function\*: Designed to minimize the error between the calculated energy requirement and the desired objective (e.g., weight maintenance, loss, or gain).
* Selection\*: Roulette wheel selection method.
* Crossover and Mutation\*: Applied to introduce variation and help the algorithm explore the solution space.
* Termination Condition\*: Either a fixed number of generations or a convergence criterion.

The fitness function plays a crucial role and is based on the Harris-Benedict equation, modified according to the individual’s lifestyle.

**GA operational flow:**

****

**11.4 System Implementation**

The implementation is carried out in a MATLAB environment, and the interface is developed for user-friendliness. Users input their details, and the GA processes the data to output the recommended calorie intake.

GUI interface showing input fields and output:

**11.5 Performance Metrics**

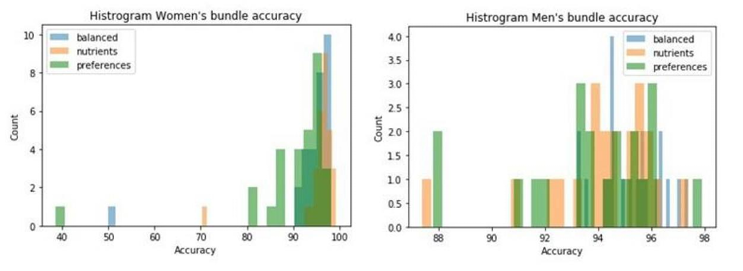
Several test cases are examined to validate the effectiveness of the GA-based system:

* Accuracy: Compared to standard manual calculators, the system shows better accuracy and adaptability.
* Time Efficiency: The algorithm converges in a reasonable number of generations, making it computationally practical.
* Personalization: Unlike static calculators, the model adapts to a user’s physical and activity profile dynamically.

**11.6 Results and Discussion**

The results show a consistent improvement in calorie prediction accuracy. GA finds optimal values that more closely align with medically and nutritionally accepted standards. The system demonstrates a superior ability to adjust to different user profiles, including overweight or underweight individuals, and those with high or low activity levels.

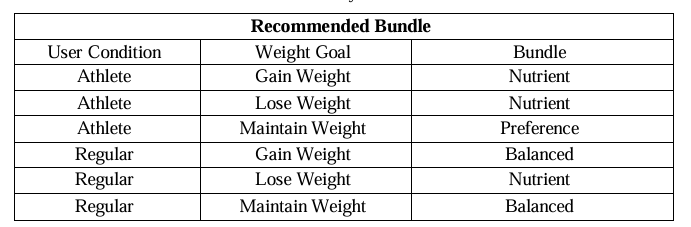
**Graph comparing GA results vs conventional calculators:**



**11.7 Case Studies**

The paper includes several user profiles to illustrate the adaptability of the system. Each profile varies in age, gender, and lifestyle, and the calorie recommendation varies accordingly, validating the flexibility of the GA approach.

**Comparison of calorie recommendations for different users:**

****

**11.8 Limitations**

While effective, the system does not yet account for more nuanced health factors such as existing medical conditions, allergies, or psychological eating behaviors. These limitations highlight the potential for future enhancements using multi-objective optimization or hybrid AI models.

**11.9 Conclusion**

The proposed Genetic Algorithm-based daily calorie recommendation system offers a **more intelligent and personalized approach** to dietary planning. It addresses the shortcomings of static calculators by incorporating evolutionary computation principles, offering better adaptability, accuracy, and user-centric recommendations.

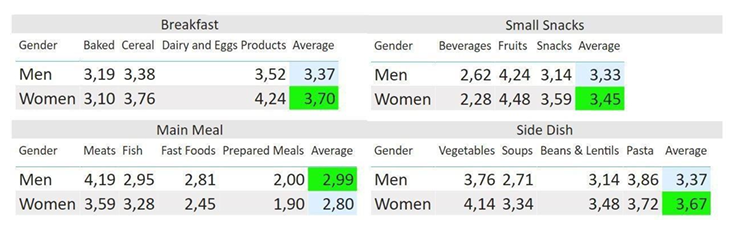
The system is scalable and can be further developed into a mobile or web application. It serves as a proof of concept for the integration of AI techniques in public health tools, especially in dietary management and lifestyle monitoring.

**11.10 Recommendations for Future Work**

Future developments could include:

* Integration of additional health parameters such as chronic conditions or blood sugar levels.
* Development of mobile or cloud-based applications for broader accessibility.
* Use of hybrid models (e.g., combining GA with neural networks) for enhanced learning and prediction.
* Inclusion of feedback mechanisms for continual user-specific calibration.

**Proposed future model architecture integrating feedback loop:**

****