Optimizing Dietary Choices

An Algorithmic Approach Towards Nutrition and Cost

Ivan Wirawan - 13523046
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: ivanwirawan25@gmail.com , 13523046@std.stei.itb.ac.id

***Abstract*—**In this era of everything become fast-paced. Society has preferred convenient and ready-to-eat meals. Yet, this diet choice is not healthy. The high number of calories and low in nutrients may lead to long-term health issues. Some people will find it complex on planning a nutritious and cost-effective diet. Therefore, this paper proposes a digital solution to this problem. An algorithmic approach to optimize dietary choices by balancing nutritional needs and financial constraints. The amount of customization will help every part of society to select the best combination of food that brings them towards their goal. This research implements and evaluates three algorithmic strategies which are greedy, dynamic programming, and branch and bound. By implementing these algorithms in a Java-based application that allows users to input available foods, nutritional information, budget and nutritional bounds, all of those algorithms were assessed. As conclusion, the performance of dynamic programming algorithms has been proved as the most stable among the three. This paper has provided solutions for the most effective and efficient method on creating personalized, healthy, and affordable meal plans.

Keywords*—Dietary Optimization, Algoritmic, Greedy, Dynamic Programming, Branch and Bound, Constraints.*

I. INTRODUCTION

A. Background

In this modern era, great nutrition has become the key to longevity. It has become one thing every person should focus on a daily basis. The importance of proper nourishment on every meal has increased in the era of instant and ready-to-eat. Due to the simplicity and cost, people have now preferred ready-to-eat meals. Yet, this approach on fulfilling the daily needs of nutrition is categorized as unhealthy. The high number of calories and lack of meaningful nutrients in instant meals will become a problem in the near future. Malnourishment and overconsumption will lead to diabetes, high cholesterol level, and many more diseases.

Despite the convenience, one of the reasons why people still choose ready-to-eat meals is the complexity of creating a whole diet menu. Simply choosing which meal to eat, which ingredients to cook has become a problem in this fast-paced living world. Moreover, determining which meals that have the suitable nutrition content towards our goal has become a greater trouble. On the side note, adjusting our diet to the financial conditions is also an important factor in creating the most sustainable diet. For some people, this has become a quite troublesome addition to their life. Therefore, they choose instant yet unhealthy food.

This judgment and decision on ready-to-eat meals is affecting the body, health, and mind badly. A balanced and nutritious diet is one of the important aspects of life. As it is crucial to maintain good health and prevent disease. Being able to provide the body with the proper macronutrients such as proteins, carbohydrates, and fats and macronutrients such as vitamins and minerals will provide the body with the necessary nutrients for growth, development, and overall well-being. Our energy levels, mood, and the ability to fight illness will be affected greatly by our diet.

Back on the problem of creating a balanced diet, it has become a headache for people to purposely direct their diet towards their goal. Such as a bodybuilder who wants to focus on their protein intake daily, or a diabetic who wants to cut off their sugar level and many more. Some people are confused about choosing the ingredients they want to cook or the meals they want to choose daily. Whether that satisfies their goal or instead ruining all over it remains one primary problem.

By recognizing this problem, society now needs a system that able to create the most suitable diet for themselves. Purposely planning on the ingredients or meals to eat daily and understanding the nutrient content is the key point. Being able to adjust the lower bound and upper bound on the micronutrients and macronutrients content that satisfies each person's goal is the feature needed. On top of that, being able to pick proper meals with the least cost among the selections will become a plus point on the system.

Therefore, on optimizing dietary choices, people need a system. This system can be achieved by utilizing algorithms that already exist on picking decisions. The capability of the algorithm to select from a lot of choices down to the perfect match in a short time will become convenient for the user. This digital system based on applications will be an algorithmic approach towards nutrition and cost.

B. Problem Statement

As for the problem statement for this research paper, it is as stated below:

1. What algorithm to utilize on creating the best choice upon considering many factors in a short time on optimizing dietary choices?
2. How to present the system in digital application that eases the user's experience on daily usage?

C. Objectives

For the objectives this research is stated below:

1. Utilizing algorithms strategy to optimize dietary choices.
2. Creating a digital system that facilitates this algorithmic approach.

D. Benefits of the Study

This study is contributing to society in many ways. The problems that are recognized and solved are critical problems in this era. Creating a system that provides a feature to optimizing dietary choices with lots of factors considered will answer the problems thoroughly. Therefore, this study is beneficial for society, for every type of person.

II. Theoretical Foundations

A. Algorithms Strategy

The definition of algorithmic strategy is an approach towards a computational problem. The approach used is an algorithmic approach which defines that they are in a structured and defined procedure manner. In this world, there exists many problems. That includes searching problems, sequencing problems, and election problems. The main objective of those problems is the ability to solve them with the best solution in the most efficient time.

With the instantiation becoming greater, the solution will be more difficult to find. Therefore, an algorithmic approach is needed. A procedure that contains step-by-step approach on solving a problem is considered an algorithm. The capability to process input and create an output that is suitable and correct on the exact problem is the key points of the algorithm.

There are lots of algorithms that already exist in this world. Few of the examples are brute force algorithm, greedy algorithm, divide and conquer, decrease and conquer, breadth first search, depth first search, backtracking, branch and bound, string matching, dynamic programming, and many more. Each of the algorithm is developed to recognize a type of problem. The performance of the algorithm is assessed by the efficiency they can provide in solving the problem. The parameter on assessing algorithms includes time and space complexity. Time complexity is the time needed with the computation procedures on processing the input into the output. Whereas space complexity is the memory space needed by the algorithm on processing the input.

Each of the algorithms provides procedures that recognize a particular type of problem or approach. Such as the brute force and greedy algorithm are considered a direct solution strategies where they asses directly the input. The state-space base strategies include backtracking and branch and bound algorithm. Top-down solution strategies that include divide and conquer, decrease and conquer, and dynamic programming.

The optimization of dietary choices can be solved with a lot of algorithms. Yet, there are several algorithms that are the most suitable for this problem. First, this problem is a decision problem. The solution includes several choices upon determining whether the meal or ingredient is picked or not. Mainly this problem can be solved with brute force algorithm. The approach on brute force algorithm is to assess every possible outcome and pick the best type. Yet, this approach is inefficient in time and space complexity. The algorithm chosen for this system should satisfy efficient time and space usage and provide an optimal solution in every case.

This problem of picking dietary choices is similar to the classic knapsack problem. The problem is which load to put into the knapsack that will give the most amount of profit in the least possible weights. In addition, this diet problem includes a lot more variables. The variables considered and priorities picked are broader and more specific than the knapsack problem. This problem offers an upper bound and lower bound for each variable. In addition, the prioritization of some variables that want to be maximized or minimized creates a great additional complexity.

On addressing such problems, there are several algorithms that are suitable. One of them is greedy algorithms. The concept of greedy algorithms is to pick a local optimum solution. Meaning in every step, the algorithm will pick what seems the best option at that time. Yet, this may not always lead to the global optimum solution, meaning the best solution for the whole case. This algorithm has a better time and space complexity than the brute force. Even though this does not guarantee an optimum solution for every case, this algorithm will create a fast approach to the problem. Another one is dynamic programming. This approach will divide the problem-solving steps into stages. The perspective towards those stages is a series of interconnected decisions. The dynamic programming approach is more effective than the greedy algorithms as it can assess more than one series of decisions in one time. The last one is branch and bound algorithms. This problem can also be solved using state space tree searching. Branch and bound algorithms are mainly used for optimization problems such as minimizing or maximizing a variable whilst not violating the constraints. The difference between other state space algorithms, the branch and bound algorithm will not do further exploration towards the node that is not leading to the solution.

These three algorithms are the chosen one upon solving this problem of dietary choices. Yet, the most effective and efficient algorithms will be proven by experimenting on multiple cases. Each of these algorithms has their own advantages and disadvantages. On solving problems, they have their own characteristics for the approach and solving methods. Therefore, to prove which algorithms is most suitable for this problem by guaranteeing an optimal solution with the least time and space complexity will be done in a series of experiments later this paper.

B. Greedy Algorithm

The greedy algorithm is the most popular and simple method on solving optimization problems. Those optimizations problems include maximization or minimization. Therefore, the main concentration of the greedy algorithms is to provide a series of decisions that optimize one of the variables. In every step, the concentration of the approach remains the same.

The main principle of greedy algorithm is "take what you can get now!". This algorithm assesses the problem step by step. In each step, there are many choices to be evaluated. In this case, is this food/ingredient suitable for the main goal? The greedy algorithm will assess a key point value on one variable that is being optimized. This will create a solution that is local optimum on every step. After that, the greedy algorithm hopes that the next series of decisions will create a global optimum solution. This approach will choose the best option for now yet neglecting the future consequences of that choice.

There are six elements of the greedy algorithm. First one is the candidate set which contains every candidate that can be chosen on every step. The solution set, which contains the candidates that have been chosen. The last four are functions that will be used to assess the value of each candidate. One function is the solution function. This function checks whether the solution set has given the solution meaning the whole problem is already solved. The selection function is used to assess each step. That function will choose the most suitable candidate according to the greedy strategy. This function tends to be heuristic. A feasible function is the function used to check whether the candidates can be put into the solution set meaning whether that is violating the constraints or not. The last one is the objective function. This function is the key function on greedy algorithms. The objective function contains the main goal for the algorithm which is maximizing or minimizing one variable.

One big disadvantage of the greedy algorithm is the inability to provide optimal solution for every case. As the algorithm only assesses each step independently, it may not lead to the best solution. The greedy algorithm does not operate completely on the solution possibility like exhaustive search. This approach may not consider every possible solution to this problem. But this approach will save the time and space needed to solve. The second reason is that the are many choices of selection functions. The implementation of this algorithm should choose the correct selection function in order to make sure that this algorithm will provide the optimal solution for the problem.

Greedy algorithms are used on many problems. The examples are coin exchange problem, activity selection problem, knapsack problem, job scheduling problem, minimum spanning tree, shortest path, Huffman code, Egyptian fraction, and many more. In all of those problems, every element of greedy algorithm is different. Those elements will be customized for the type and the criteria of every problem. On those problems, greedy algorithms may or may not provide the optimal solution.

C. Dynamic Programming Algorithm

The usage of dynamic programming is quite like a greedy algorithm. Which is the optimization problem, whether to maximize or minimize the value of one variable. Yet, the approach of dynamic programming is completely different. This method will divide the whole solution into a group of stages. In each stage, it contains a decision. As the whole stages are combined, the solution will be a series of decisions that are interconnected. The main difference of dynamic programming from the greedy algorithms is the capability of assessing more than one series of decisions in one time.

The dynamic programming approach is based on optimality principle. Which is defines as "if the total solution is optimal, therefore the parts of the solution is also optimal." The main characteristic of dynamic programming is the problems are divided into several stages. On each stage, there is a decision to be chosen. Each stage will consist of several states that will be interconnected on that stage. The definition of status is the various possible inputs that exist at one stage. The solution of the decision in each stage will be transformed to the next status in the next stage. The cost of each stage will increase steadily with the increase of the steps. This algorithm also assesses the cost on each stage with the cost from previous stages and the cost needed for the next stage. This recursive relationship will identify the best decision on every status that will lead to the best decision on the next status. This is the utilization of optimality principles.

The procedure on using dynamic programming is to structure the optimal solution characteristic. This includes the steps, decision variables, states, and many more. The next step is to define the recursive relationship between the optimal solutions on each stage. After defining that relationship, the optimal solution can be calculated by a forward or backward approach. The last one is to reconstruct the solution in order to present the series of decisions as a whole.

Dynamic programming is also commonly used on many problems. Such problems are shortest path, integer knapsack, capital budgeting, travelling salesman problem, and many more. Those problems are similar to the dietary choices problem. That is the reason why dynamic programming is chosen to solve this problem.

D. Branch and Bound Algorithm

The branch and bound algorithm are also commonly used on optimization problems. The key difference between the algorithms that have been reviewed until now is that this algorithm utilizes state-space searching. The other examples of state-space searching algorithms include breadth first search, depth first search, depth-limited search, backtracking, and many more. The concept of branch and bound algorithm is a combination of breadth first search and least cost search. Meaning on every exploration of a node, the node chosen is with the least cost on minimization case.

The nature of branch and bound algorithms is quite similar with backtracking. The solution searching by utilizing state-space tree is shared between the two. The main concept of not exploring the node that seems to off lead from the objective of

the problem is also shared. Yet the key differences include the usage of backtracking on mainly non-optimization problem and branch and bound on optimization problems. Also, in the branch and bound algorithm there exists a determination of bounding function on each solution. The last difference is on node generation criteria, the backtracking algorithm uses depth first search concept. While branch and bound algorithms use several criteria upon assessing the node. The algorithm will determine the best node to explore.

The branch and bound algorithm also have a bounding function. This function will prune the path that is assumed to not lead to the optimum solution anymore. The criteria of the pruning function are that the node value is not better than the best solution so far and the solution does not represent a feasible decision as it violates the constraints.

On exploring the node in tree, every node is given a cost value. That cost represents the estimation of the least cost path to the solution status node. The next node to be expanded will be chosen by the least cost search or most cost search in accordance with the objective. This effectively prunes the searching space therefore cuts the time and space complexity.

III. Implementation Design

A. Overview

The solution approach towards optimizing dietary choices problem is successfully implemented on a digital platform. The application was built on Java programming language with the help of Java Swing on building the Graphical User Interface (GUI). This program will help users to find the optimal combination of foods or ingredients that meet nutritional requirements while staying within budget constraints. In order to test which algorithm suits this problem better, the program offers 3 different algorithms strategy that includes greedy algorithms, branch and bound, and dynamic programming algorithms. Each of these algorithms has the goal towards optimizing one variable in the problem. Yet, the approach of each algorithm is different than each other, making the time and space complexity completely different.

Built on top of Object-Oriented-Programming (OOP), this application was built on top of collaborating classes. The main class in the Food class represents individual food items with nutrients content such as protein, fat carbohydrates, and many more. This is the class that will hold all the information regarding the candidates for the solution (food or ingredient). The next class is NutritionalConstraints class which defines the minimum nutritional requirements. This class also holds the user input for all the constraints that are defined. It can include the minimum number of proteins, fat, or the maximum number of monies, and prioritization. The last main class is OptimizationResult which contains the selected foods, total value, total cost, and algorithm used for the solution.

By interacting with GUI, the user can input all of the variables needed. In the first page, user can input the choices of foods or ingredients that is available to them. This includes their nutrition contents such as food name, protein, fat, carbohydrates, vitamins and its price. Users can add any number of choices that will be assessed later. The GUI will also show all the data that has been input in the form of tables. In the optimization page, the user can input the minimum nutritional requirements, budget, priority, and algorithm. This is where the user can define all of the constraint necessary upon their specific problem. It is fully customizable by the user according to each of one's goal and problems. The user can also choose which variables to prioritize between all of the variables available. After finishing all the data, user can start the optimization algorithm by clicking the "RUN OPTIMIZATION" button. The user will be directed to the results page which contains all the information about the solution. Whether there is a solution or not, the program will show all the details on that. The GUI will display the selected foods, the summary of total cost and nutrition content, and whether that contents satisfy the constraints. This will give the user a brief yet detailed explanation upon which food/ingredient to choose and process. On assessing the efficiency and effectiveness of all the algorithms, the GUI will also state the execution time of the process. With the less time needed, the algorithm has proved to be better and faster.

B. Algorithm Implementation

As stated before, there are 3 different algorithms that is implemented on this program which are greedy algorithm, dynamic programming algorithms, and branch and bound algorithms. Every algorithm has a different strategy and process upon solving the problem. This is due to the characteristics of every algorithm. Yet, the input and output remain the same. The input includes a list of foods, optimization priority, budget limit, and nutritional constraints. While the output contains selected foods, total value, total cost, and the algorithm used.

The greedy algorithm implemented has the strategy of constraint-first greedy approach. By using that approach, its process includes sorting foods by value-to-price ratio while prioritizing constraint satisfactions first. If the budget still remains, additional food will be added. This algorithm is focused on meeting the nutritional requirements and will do the sorting later. The criterion for sorting is prioritization of most values per money. The time complexity of this algorithm is O(n2) with n is the number of choices. With this algorithm, the advantage is fast execution and a guaranteed constraints-satisfaction solution. But it may not find the optimal global solution for this problem.

Greedy algorithm is created on top of their elements. The candidate set here includes all available food items that have not been selected yet. The solution set contains the currently selected foods that form partial/complete solution. As for the selection function, it is divided into 2 phases. The first phase selects food that maximizes constraints improvement, and the second phase will select foods with the highest value-to-price ratio. This means that the solution function for first phase is when the nutritional constraints are satisfied and the second phase is when the budget is exhausted or no more food available. The main bounding or feasible function of this

algorithm is whether the food choices have surpassed the budget constraints. That constraint is the main constraint of this problem. The last function, which is the objective function, also differs on each phase. The first phase maximizes constraint satisfaction and phase 2 maximizes total value on the priority criteria.

The dynamic programming algorithm has a strategy of two-phase optimization. In the first phase, this algorithm will find constraint-satisfying solutions. After that, the second phase will begin which includes using standard knapsack dynamic programming approach on the remaining items. This two-phase hybrid approach combining constraint satisfaction with an optimal knapsack solution will guarantee the optimum solution for this problem. It has the time complexity of O(n x budget). The advantage of this algorithm is the capability of finding optimal solution for the knapsack portion. Yet, the disadvantage is high time and space usage on large budgets. On this approach, the candidates are selected by their cost and value. The state exploration cost is the maximum achievable value with food and budget. This will maximize the total value while staying within the budget.

Aside from the two algorithms, the branch and bound algorithms use systematic tree search with pruning strategy. The process includes the priority queue for best-first search. While doing that search, the algorithm also calculates the upper bound for pruning and eliminating solution that is not feasible. After that, this algorithm will sort foods by value-to-price ratio and presenting it as the solution. In the state-space, the direct cost of the node is sum of current price (that has already been picked) with the next food price. The constraint is that it is not to exceed the budget. The cost-benefit analysis will create an output of maximum possible additional value achievable and is used upon deciding whether to explore or prune branches as the bounding function. The advantage of this approach is the intelligent pruning used. This will cause a decrease in time and space complexity. But, on a large-scale problem, without a good bounding function, the algorithm can take a lot of time.

IV. Result and Analysis

A. *Result on Test Cases*

1. Test Case 1

   INFORMATION:

   Objectives: Maximizing protein intake with limited cost.

   List of Foods: (Name, Protein, Fat, Carbs, Vitamin, Price)
   - Tuna, 25, 5, 0, 10, 25000
   - Chicken Breast, 30, 3, 0, 8, 28000
   - Lentils, 18, 1, 30, 5, 12000
   - Greek Yogurt, 20, 10, 8, 2, 18000
   - Tofu, 15, 8, 5, 3, 15000
   - Salmon, 22, 12, 0, 15, 35000

   Budget: 50000

   Constraints: (minimal)
   - Protein: 40
   - Fat: 10
   - Carbs: 10
   - Vitamin: 8

   Priority: Protein

   RESULTS:

   - Greedy Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Greedy Algorithm
Execution Time: 5 ms
==================================================

Nutritional Constraints: Min Protein: 40.0g, Min Fat: 10.0g, Min Carbohydrate: 10.0g, Min Vitamin: 8.0mg

Selected Foods:
1. Chicken Breast (Protein: 30.0, Fat: 3.0, Carbohydrate: 0.0, Vitamin: 8.0, Price: 28000.00)
2. Greek Yogurt (Protein: 20.0, Fat: 10.0, Carbohydrate: 8.0, Vitamin: 2.0, Price: 18000.00)

Summary:
Total Optimal Value: 50.00
Total Cost: Rp 46000.00

Total Nutrition:
- Protein: 50.0 g [OK]
- Fat: 13.0 g [OK]
- Carbohydrate: 8.0 g [X]
- Vitamin: 10.0 mg [OK]

Constraint Status: NOT SATISFIED [X]

==================================================
```

   - Dynamic Programming Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Dynamic Programming
Execution Time: 14 ms
==================================================

Nutritional Constraints: Min Protein: 40.0g, Min Fat: 10.0g, Min Carbohydrate: 10.0g, Min Vitamin: 8.0mg

Selected Foods:
1. Lentils (Protein: 18.0, Fat: 1.0, Carbohydrate: 30.0, Vitamin: 5.0, Price: 12000.00)
2. Greek Yogurt (Protein: 20.0, Fat: 10.0, Carbohydrate: 8.0, Vitamin: 2.0, Price: 18000.00)
3. Tofu (Protein: 15.0, Fat: 8.0, Carbohydrate: 5.0, Vitamin: 3.0, Price: 15000.00)

Summary:
Total Optimal Value: 53.00
Total Cost: Rp 45000.00

Total Nutrition:
- Protein: 53.0 g [OK]
- Fat: 19.0 g [OK]
- Carbohydrate: 43.0 g [OK]
- Vitamin: 10.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

   - Branch and Bound Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Branch and Bound
Execution Time: 2 ms
==================================================

Nutritional Constraints: Min Protein: 40.0g, Min Fat: 10.0g, Min Carbohydrate: 10.0g, Min Vitamin: 8.0mg

Selected Foods:
1. Lentils (Protein: 18.0, Fat: 1.0, Carbohydrate: 30.0, Vitamin: 5.0, Price: 12000.00)
2. Chicken Breast (Protein: 30.0, Fat: 3.0, Carbohydrate: 0.0, Vitamin: 8.0, Price: 28000.00)

Summary:
Total Optimal Value: 48.00
Total Cost: Rp 40000.00

Total Nutrition:
- Protein: 48.0 g [OK]
- Fat: 4.0 g [X]
- Carbohydrate: 30.0 g [OK]
- Vitamin: 13.0 mg [OK]

Constraint Status: NOT SATISFIED [X]

==================================================
```

2. Test Case 2

   INFORMATION:

   Objectives: Achieving balanced nutrition with a moderate budget, focusing on carbohydrates.

   List of Foods: (Name, Protein, Fat, Carbs, Vitamin, Price)
   - Brown Rice, 8, 2, 45, 2, 10000
   - Quinoa, 14, 6, 40, 4, 22000
   - Sweet Potato, 4, 0, 26, 10, 8000
   - Avocado, 4, 20, 15, 6, 16000
   - Spinach, 5, 0, 7, 12, 5000
   - Almonds, 6, 14, 6, 3, 19000

Budget: 60000
Constraints: (minimal)
- Protein: 20
- Fat: 25
- Carbs: 70
- Vitamin: 20

Priority: Carbohydrate

RESULTS:
- Greedy Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Greedy Algorithm
Execution Time: 0 ms
==================================================

Nutritional Constraints: Min Protein: 20.0g, Min Fat: 25.0g, Min Carbohydrate: 70.0g, Min Vitamin: 20.0m

Selected Foods:
1. Quinoa (Protein: 14.0, Fat: 6.0, Carbohydrate: 40.0, Vitamin: 4.0, Price: 22000.00)
2. Avocado (Protein: 4.0, Fat: 20.0, Carbohydrate: 15.0, Vitamin: 6.0, Price: 16000.00)
3. Sweet Potato (Protein: 4.0, Fat: 0.0, Carbohydrate: 26.0, Vitamin: 10.0, Price: 8000.00)
4. Brown Rice (Protein: 8.0, Fat: 2.0, Carbohydrate: 45.0, Vitamin: 2.0, Price: 10000.00)

Summary:
Total Optimal Value: 126.00
Total Cost: Rp 56000.00

Total Nutrition:
- Protein: 30.0 g [OK]
- Fat: 28.0 g [OK]
- Carbohydrate: 126.0 g [OK]
- Vitamin: 22.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

- Dynamic Programming Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Dynamic Programming
Execution Time: 2 ms
==================================================

Nutritional Constraints: Min Protein: 20.0g, Min Fat: 25.0g, Min Carbohydrate: 70.0g, Min Vitamin: 20.0m

Selected Foods:
1. Brown Rice (Protein: 8.0, Fat: 2.0, Carbohydrate: 45.0, Vitamin: 2.0, Price: 10000.00)
2. Sweet Potato (Protein: 4.0, Fat: 0.0, Carbohydrate: 26.0, Vitamin: 10.0, Price: 8000.00)
3. Spinach (Protein: 5.0, Fat: 0.0, Carbohydrate: 7.0, Vitamin: 12.0, Price: 5000.00)
4. Avocado (Protein: 4.0, Fat: 20.0, Carbohydrate: 15.0, Vitamin: 6.0, Price: 16000.00)
5. Almonds (Protein: 6.0, Fat: 14.0, Carbohydrate: 6.0, Vitamin: 3.0, Price: 19000.00)

Summary:
Total Optimal Value: 99.00
Total Cost: Rp 58000.00

Total Nutrition:
- Protein: 27.0 g [OK]
- Fat: 36.0 g [OK]
- Carbohydrate: 99.0 g [OK]
- Vitamin: 33.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

- Branch and Bound Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Branch and Bound
Execution Time: 0 ms
==================================================

Nutritional Constraints: Min Protein: 20.0g, Min Fat: 25.0g, Min Carbohydrate: 70.0g, Min Vitamin: 20.0m

Selected Foods:
1. Brown Rice (Protein: 8.0, Fat: 2.0, Carbohydrate: 45.0, Vitamin: 2.0, Price: 10000.00)
2. Sweet Potato (Protein: 4.0, Fat: 0.0, Carbohydrate: 26.0, Vitamin: 10.0, Price: 8000.00)

Summary:
Total Optimal Value: 71.00
Total Cost: Rp 18000.00

Total Nutrition:
- Protein: 12.0 g [X]
- Fat: 2.0 g [X]
- Carbohydrate: 71.0 g [OK]
- Vitamin: 12.0 mg [X]

Constraint Status: NOT SATISFIED [X]

==================================================
```

3. Test Case 3

INFORMATION:
Objectives: Maximizing vitamin intake with a generous budget.
List of Foods: (Name, Protein, Fat, Carbs, Vitamin, Price)
- Orange, 1, 0, 12, 53, 6000
- Bell Pepper, 1, 0, 6, 128, 7000
- Broccoli, 3, 0, 6, 89, 9000
- Carrots, 1, 0, 10, 41, 4000
- Kale, 4, 1, 9, 60, 5000
- Kiwi, 1, 1, 15, 93, 8000

Budget: 35000
Constraints: (minimal)
- Protein: 5
- Fat: 1
- Carbs: 30
- Vitamin: 200

Priority: Vitamin

RESULTS:
- Greedy Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Greedy Algorithm
Execution Time: 1 ms
==================================================

Nutritional Constraints: Min Protein: 5.0g, Min Fat: 1.0g, Min Carbohydrate: 30.0g, Min Vitamin: 200.0mg

Selected Foods:
1. Bell Pepper (Protein: 1.0, Fat: 0.0, Carbohydrate: 6.0, Vitamin: 128.0, Price: 7000.00)
2. Kiwi (Protein: 1.0, Fat: 1.0, Carbohydrate: 15.0, Vitamin: 93.0, Price: 8000.00)
3. Kale (Protein: 4.0, Fat: 1.0, Carbohydrate: 9.0, Vitamin: 60.0, Price: 5000.00)
4. Carrots (Protein: 1.0, Fat: 0.0, Carbohydrate: 10.0, Vitamin: 41.0, Price: 4000.00)
5. Broccoli (Protein: 3.0, Fat: 0.0, Carbohydrate: 6.0, Vitamin: 89.0, Price: 9000.00)

Summary:
Total Optimal Value: 411.00
Total Cost: Rp 33000.00

Total Nutrition:
- Protein: 10.0 g [OK]
- Fat: 2.0 g [OK]
- Carbohydrate: 46.0 g [OK]
- Vitamin: 411.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

- Dynamic Programming Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Dynamic Programming
Execution Time: 47 ms
==================================================

Nutritional Constraints: Min Protein: 5.0g, Min Fat: 1.0g, Min Carbohydrate: 30.0g, Min Vitamin: 200.0mg

Selected Foods:
1. Bell Pepper (Protein: 1.0, Fat: 0.0, Carbohydrate: 6.0, Vitamin: 128.0, Price: 7000.00)
2. Kale (Protein: 4.0, Fat: 1.0, Carbohydrate: 9.0, Vitamin: 60.0, Price: 5000.00)
3. Kiwi (Protein: 1.0, Fat: 1.0, Carbohydrate: 15.0, Vitamin: 93.0, Price: 8000.00)
4. Carrots (Protein: 1.0, Fat: 0.0, Carbohydrate: 10.0, Vitamin: 41.0, Price: 4000.00)
5. Orange (Protein: 1.0, Fat: 0.0, Carbohydrate: 12.0, Vitamin: 53.0, Price: 6000.00)

Summary:
Total Optimal Value: 375.00
Total Cost: Rp 30000.00

Total Nutrition:
- Protein: 8.0 g [OK]
- Fat: 2.0 g [OK]
- Carbohydrate: 52.0 g [OK]
- Vitamin: 375.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

- Branch and Bound Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Branch and Bound
Execution Time: 1 ms
==================================================

Nutritional Constraints: Min Protein: 5.0g, Min Fat: 1.0g, Min Carbohydrate: 30.0g, Min Vitamin: 200.0mg

Selected Foods:
1. Bell Pepper (Protein: 1.0, Fat: 0.0, Carbohydrate: 6.0, Vitamin: 128.0, Price: 7000.00)
2. Kale (Protein: 4.0, Fat: 1.0, Carbohydrate: 9.0, Vitamin: 60.0, Price: 5000.00)
3. Kiwi (Protein: 1.0, Fat: 1.0, Carbohydrate: 15.0, Vitamin: 93.0, Price: 8000.00)

Summary:
Total Optimal Value: 281.00
Total Cost: Rp 20000.00

Total Nutrition:
- Protein: 6.0 g [OK]
- Fat: 2.0 g [OK]
- Carbohydrate: 30.0 g [OK]
- Vitamin: 281.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

4. Test Case 4

INFORMATION:

Objectives: Finding the cheapest food combination that meets high nutritional demands.

List of Foods: (Name, Protein, Fat, Carbs, Vitamin, Price)

- Eggs, 12, 10, 1, 6, 15000
- Oats, 16, 6, 65, 4, 18000
- Peanut Butter, 8, 16, 20, 2, 22000
- Milk, 8, 8, 12, 10, 13000
- Banana, 1, 0, 28, 9, 7000
- Beans, 15, 1, 25, 5, 11000

Budget: 70000

Constraints: (minimal)

- Protein: 40
- Fat: 20
- Carbs: 80
- Vitamin: 15

Priority: Price

RESULTS:

- Greedy Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Greedy Algorithm
Execution Time: 0 ms
==================================================

Nutritional Constraints: Min Protein: 40.0g, Min Fat: 20.0g, Min Carbohydrate: 80.0g, Min Vitamin: 15.0m

Selected Foods:
1. Oats (Protein: 16.0, Fat: 6.0, Carbohydrate: 65.0, Vitamin: 4.0, Price: 18000.00)
2. Peanut Butter (Protein: 8.0, Fat: 16.0, Carbohydrate: 20.0, Vitamin: 2.0, Price: 22000.00)
3. Beans (Protein: 15.0, Fat: 1.0, Carbohydrate: 25.0, Vitamin: 5.0, Price: 11000.00)
4. Milk (Protein: 8.0, Fat: 8.0, Carbohydrate: 12.0, Vitamin: 10.0, Price: 13000.00)

Summary:
Total Optimal Value: 0.00
Total Cost: Rp 64000.00

Total Nutrition:
- Protein: 47.0 g [OK]
- Fat: 31.0 g [OK]
- Carbohydrate: 122.0 g [OK]
- Vitamin: 21.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

- Dynamic Programming Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Dynamic Programming
Execution Time: 9 ms
==================================================

Nutritional Constraints: Min Protein: 40.0g, Min Fat: 20.0g, Min Carbohydrate: 80.0g, Min Vitamin: 15.0m

Selected Foods:
1. Oats (Protein: 16.0, Fat: 6.0, Carbohydrate: 65.0, Vitamin: 4.0, Price: 18000.00)
2. Beans (Protein: 15.0, Fat: 1.0, Carbohydrate: 25.0, Vitamin: 5.0, Price: 11000.00)
3. Milk (Protein: 8.0, Fat: 8.0, Carbohydrate: 12.0, Vitamin: 10.0, Price: 13000.00)
4. Eggs (Protein: 12.0, Fat: 10.0, Carbohydrate: 1.0, Vitamin: 6.0, Price: 15000.00)

Summary:
Total Optimal Value: 0.00
Total Cost: Rp 57000.00

Total Nutrition:
- Protein: 51.0 g [OK]
- Fat: 25.0 g [OK]
- Carbohydrate: 103.0 g [OK]
- Vitamin: 25.0 mg [OK]

Constraint Status: SATISFIED [OK]

==================================================
```

- Branch and Bound Algorithm:

```
==================================================
OPTIMIZATION RESULTS - Branch and Bound
Execution Time: 0 ms
==================================================

Nutritional Constraints: Min Protein: 40.0g, Min Fat: 20.0g, Min Carbohydrate: 80.0g, Min Vitamin: 15.0m

Selected Foods:
1. Beans (Protein: 15.0, Fat: 1.0, Carbohydrate: 25.0, Vitamin: 5.0, Price: 11000.00)
2. Milk (Protein: 8.0, Fat: 8.0, Carbohydrate: 12.0, Vitamin: 10.0, Price: 13000.00)

Summary:
Total Optimal Value: 0.00
Total Cost: Rp 24000.00

Total Nutrition:
- Protein: 23.0 g [X]
- Fat: 9.0 g [X]
- Carbohydrate: 37.0 g [X]
- Vitamin: 15.0 mg [OK]

Constraint Status: NOT SATISFIED [X]

==================================================
```

B. Result Analysis

Based on the four test cases that have been done, the performance of each algorithm can be assessed with criteria such as ability to generate the optimal solution, cost-effectiveness, constraint satisfaction, and time complexity. On those criteria, it can be determined which algorithm is the most suitable approach to this problem.

In the first test case, with the objective of maximizing protein intake. This is commonly used diet by bodybuilders. In terms of solution, the dynamic programming algorithm is able to pick the highest protein value within the budget which is 53 grams. Yet, this method has the most amount of time consumed upon solving. But, as the time consumed is still considered fast, dynamic programming is still a great solution in this test case.

For the second test case, the greedy algorithm was able to achieve the highest carbohydrate value of 126 while the dynamic programming approach solution achieves a carbohydrate value of 93 grams. Notably, the branch and bound algorithm failed to satisfy the constraints. This gave significant signs of the inability of branch and bound upon complex problems.

For the third case, with the goal to maximize vitamin variable, the branch and bound algorithm successfully provided the best solution. The solution provided is 281 milligrams of vitamin on the cost of Rp. 20.000. Followed by the dynamic programming algorithm with 275 milligrams for Rp. 26.000. On simpler problem, the branch and bound can choose the best solution.

The last test is one of the complex test cases which consists of finding food combinations that met high nutritional demands on the constraints of cost. In this test, dynamic programming algorithms have proved their capability to be the most successful. It is able to satisfy all the constraints with the lowest cost of Rp. 57.000. Then, the greedy algorithm was able to meet all the constraints with a higher cost of Rp. 64.000. The branch and bound algorithm failed to provide a solution that satisfies the nutritional requirements.

Overall, the greedy algorithm consistently is always the fastest on providing a solution. While dynamic programming and branch and bound had longer execution time. This is due to the complexity of the procedure and the number of iterations done by each approach that differs.

With all of the results assessed, this analysis shows that the best algorithm to be applied on this problem is dynamic programming. On several test cases that has been done, this algorithm tends to be the most reliable and able to provide optimal solutions, especially on cost priority and high-demand constraints. Even though the greedy algorithm is the fastest, it may not always find the best solution for the problem. Even though theoretically the branch and bound algorithm can find optimal solutions, it failed in complex cases. On several occasions, the branch and bound algorithm are proved inconsistent. In a complex constraint's environment or

problem, this algorithm fails to even meet the basic constraints that are defined.

V. Conclusion

This research has successfully developed a digital system that answers the problems of optimizing dietary choices based on nutritional goals and budget limitations. The approach used was an algorithmic approach. Three algorithms that are commonly used are implemented and tested in this research. Those algorithms are greedy, dynamic programming, and branch and bound algorithms. This paper demonstrates the effectiveness of each algorithm with the test cases and its result.

After assessing the performance of each algorithm, the most suitable algorithm for this problem is the dynamic programming algorithm. While greedy algorithms prove to be the fastest, the approach used on greedy algorithms does not always guarantee an optimum solution. The branch and bound algorithm have proved its capability on specific optimizations problem. Yet, it has always failed to react to complex constraints in a problem. The inconsistency makes it less dependable on general usage of this algorithm. The best algorithm is appointed to the dynamic programming algorithm.

On multifaceted problems such as dietary planning, the dynamic programming algorithm is the recommended choice. The hybrid approach on selecting the most optimum solution will guarantee the optimality and effectiveness of time and space. On prioritizing nutritional contents and cost-efficiency, the dynamic algorithm has been proven effective.

By implementing those algorithms into digital applications, the solution is available for a greater mass. Ready and convenient to use. Other than that, this study also highlights the significant potential of algorithmic strategies on contributing to the people by optimizing dietary choices, and many other choices. Upon gaining the capability of algorithmic approaches and digital technologies, there are lots of human's daily problems that can be solved easily.

Video Link at Youtube

https://youtu.be/Wa_F_fKfmIo

Repository Link at GitHub

https://github.com/ivan-wirawan/DietaryOptimization

Acknowledgment

With this paper has been successfully completed, I want to acknowledge and express my sincere thanks to Mrs. Nur Ulfa Maulidevi, the lecturer of Algorithmic Strategy. The amount of knowledge and experience that is gained through the lectures has a great positive impact on creating this research paper.

Other than Mrs. Nul Ulfa Maulidevi, I want to express great thanks to Mr. Rinaldi Muni. Whose website is full of this paper's materials. Guiding us on every project, examinations, and many more in this Algorithmic Strategy class. The ability to access the material freely has become very convenient for us students to deepen our understanding of this class.

References

[1] Munir, R. (2025). Bahan Kuliah IF2211 Strategi Algoritma. Program Studi Teknik Informatika, Institut Teknologi Bandung.

[2] Stinson, J. P., & Guley, H. M. (1982). Use of a branch and bound algorithm to schedule food production in a semi-conventional food service system. Journal of the American Dietetic Association, 81(5), 562–567.

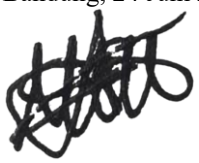
[3] Fomekong-Nanfack, Y., & Kaoutar, F. (2023). A Personalized Flexible Meal Planning for Individuals With Diet-Related Health Concerns: System Design and Feasibility Validation Study. JMIR Formative Research, 7(1), e46434.

[4] Basargan, A., & Calinov, I. C. (2021). A Hybrid Approach for Solving the Diet Problem. Applied Sciences, 11(15), 6891.

DECLARATION

Hereby, I declare that this paper I have written is my own work, not a reproduction or translation of someone else's paper and not plagiarized.

Bandung, 24 Juni 2025



Ivan Wirawan
13523046