2013-01-01

Solving Logistic System Design Problem Considering Various Biomass Feedstock Using Two Metaheuristic Optimization Methods

Jesusita Ibarra
University of Texas at El Paso, jibarra5@miners.utep.edu

Follow this and additional works at: https://digitalcommons.utep.edu/open_etd
Part of the Engineering Commons

Recommended Citation
https://digitalcommons.utep.edu/open_etd/1843

This is brought to you for free and open access by DigitalCommons@UTEP. It has been accepted for inclusion in Open Access Theses & Dissertations by an authorized administrator of DigitalCommons@UTEP. For more information, please contact lweber@utep.edu.
SOLVING LOGISTIC SYSTEM DESIGN PROBLEM CONSIDERING VARIOUS BIOMASS FEEDSTOCKS USING TWO METAHEURISTIC OPTIMIZATION METHODS

JESUSITA IBARRA
Department of Industrial, Manufacturing & Systems Engineering

APPROVED:

__________________________________________
Heidi A. Taboada, Ph.D., Chair

__________________________________________
Jose F. Espiritu, Ph.D.

__________________________________________
Paras Mandal, Ph.D.

__________________________________________
Noe Vargas-Hernandez, Ph.D.

__________________________________________
Benjamin C. Flores, Ph.D.
Dean of the Graduate School
DEDICATION

This work is dedicated to my husband, family, and friends for all their unconditional support.
SOLVING LOGISTIC SYSTEM DESIGN PROBLEM CONSIDERING VARIOUS BIOMASS FEEDSTOCK USING TWO METAHEURISTIC OPTIMIZATION METHODS

by

JESUSITA IBARRA, B.S.I.E.

THESIS

Presented to the Faculty of the Graduate School of
The University of Texas at El Paso
in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

Department of Industrial, Manufacturing & Systems Engineering
THE UNIVERSITY OF TEXAS AT EL PASO
August 2013
ACKNOWLEDGEMENTS

This thesis would not have been possible without the help, support, and patience of my advisor Dr. Heidi A. Taboada, not to mention her advice, patience, and incomparable knowledge of optimization methods, just to mention a few.
ABSTRACT

Bioenergy has become an important source of energy; it can be used as an alternative to fossil fuel energy, and it offers significant potential to alleviate climate change by reducing greenhouse gas emissions caused by the burning of fossil fuels. The Energy Independence and Security Act decrees the use of 21 billion gallons of advanced biofuel including 16 billion gallons of cellulosic biofuels by the year 2022. It is easy to observe that biomass can make a considerable contribution meet the energy demands. On the other hand, the supply of sustainable energy is one of the main challenges that must be met in the coming years if biomass is to alleviate the reliance on fossil fuels. In many ways, biomass is a unique renewable resource because in comparison to other renewable energy options, biomass can be easily stored and transported. This thesis presents two different models for the design optimization of the life-cycle of biomass logistics system through bio-inspired metaheuristic optimization considering multiple types of feedstocks. This work compares the performance and solutions obtained by two types of metaheuristic approaches: genetic algorithm and bee colony optimization. Compared to precise mathematical optimization methods, metaheuristics does not guarantee that a global optimal solution can be found on some types of problems. Similar problems to the one presented in this thesis have been previously solved using linear programming, mixed integer linear programming, and mixed integer programming methods. However, depending on the type of problem, these mathematical methods might require exponential computation time, which can result prices that are too high for practical purposes. Therefore, this thesis develops two types of metaheuristic approaches for the design optimization of the life cycle logistics system considering multiple types of feedstocks.
# TABLE OF CONTENTS

**ACKNOWLEDGEMENTS**.......................................................................................... v

**ABSTRACT** ........................................................................................................ vi

**TABLE OF CONTENTS** ..................................................................................... vii

**LIST OF TABLES** .............................................................................................. x

**LIST OF FIGURES** ............................................................................................. xi

**CHAPTER 1: INTRODUCTION** ............................................................................. 1

1.1 Background and research motivation ............................................................. 1

1.2 Thesis Objective ............................................................................................... 4

1.3 Scope and Limitations ...................................................................................... 5

1.4 Thesis Outline .................................................................................................. 6

**CHAPTER 2: LITERATURE REVIEW** ................................................................. 8

2.1 Optimization Methods .................................................................................... 8

2.1.1 Linear Methods .......................................................................................... 8

2.1.2 Mixed Integer Linear Programming Methods ......................................... 9

2.1.3 Metaheuristic Methods ............................................................................ 12

2.2 Summary ........................................................................................................ 14

**CHAPTER 3: LOGISTIC SYSTEM DESIGN PROBLEM** ................................... 15

3.1 Problem Description ....................................................................................... 15

3.2 Problem Assumptions .................................................................................... 16

3.3 Data ................................................................................................................ 17

3.4 Model ............................................................................................................. 19

3.5 Summary ........................................................................................................ 26

**CHAPTER 4: METAHEURISTIC ALGORITHMS** .............................................. 27

4.1 Genetic Algorithm .......................................................................................... 27

4.1.1 Pseudo-Code ............................................................................................ 28

4.1.2 Chromosome Implementation .................................................................. 30

4.1.2.1 Binary Encoding ................................................................................ 30

4.1.2.2 Permutation Encoding ...................................................................... 30

4.1.2.3 Value Encoding .................................................................................. 31

4.1.2.4 Tree Encoding ................................................................................... 31
## 4.1 Genetic Algorithm Pseudo-code

### 4.1.1 Crossover Operators
- 4.1.1.1 Single Point Crossover
- 4.1.1.2 Two Point Crossover
- 4.1.1.3 Multi-point Crossover

### 4.1.2 Mutation Techniques
- 4.1.2.1 Simple Mutation
- 4.1.2.2 Order-based Mutation

### 4.1.3 Selection Methods
- 4.1.3.1 Rank Selection
- 4.1.3.2 Roulette Wheel Selection
- 4.1.3.3 Tournament Selection

### 4.1.4 Crossover Algorithms
- 4.1.4.1 Simple Crossover
- 4.1.4.2 Tournament Crossover
- 4.1.4.3 Elitism
- 4.1.4.4 Steady State Selection
- 4.1.4.5 Population Size

### 4.1.5 Parameter Values
- 4.1.5.1 Population Size
- 4.1.5.2 Crossover Rate
- 4.1.5.3 Mutation Rate

### 4.1.6 Applications
- 4.1.6.1 Optimization Techniques
- 4.1.6.2 Behavioral Pseudocode
- 4.1.6.3 Steady State Pseudocode

## 4.2 BEES’ ARTIFICIAL BEHAVIOR

### 4.2.1 Natural Behavior
- 4.2.1.1 foraging
- 4.2.1.2 scout bees
- 4.2.1.3 onlooker bees

### 4.2.2 Artificial Behavior
- 4.2.2.1 construction
- 4.2.2.2 foraging
- 4.2.2.3 pheromone

## 4.3 BCO ALGORITHM PSEUDO-CODE

### 4.3.1 Pseudo-code
- 4.3.1.1 Initialization
- 4.3.1.2 Update pheromone
- 4.3.1.3 Update position

## 4.4 Numerical Results and Discussion

### 4.4.1 Sensitivity Analysis
- 4.4.1.1 Parameter Sensitivity
- 4.4.1.2 Comparison

### 4.4.2 Numerical Results
- 4.4.2.1 Comparison
- 4.4.2.2 Analysis
6.4 BCO Sensitivity Analysis ................................................................. 69
6.5 GA and BCO Comparison ............................................................... 70
6.6 Summary ......................................................................................... 71
CHAPTER 7: CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK ........72
  7.1 General .......................................................................................... 72
  7.2 Conclusions ................................................................................... 73
  7.3 Contributions ................................................................................ 73
  7.4 Recommendations for Future Work ............................................. 74
REFERENCES ......................................................................................... 75
APPENDIX A ......................................................................................... 80
APPENDIX B ......................................................................................... 104
VITA ....................................................................................................... 151
LIST OF TABLES

Table 1: Algorithms inspired by bees’ behavior.................................................................48
Table 2: Chromosome description..........................................................................................53
Table 3: BCO path options .....................................................................................................59
Table 4: $P_{follow}$ adjustment. .............................................................................................62
Table 5: GA results: costs, revenue, and profit . ....................................................................64
Table 6: GA sensitivity analysis. .............................................................................................67
Table 7: BCO results: costs, revenue, and profit .................................................................68
Table 8: BCO sensitivity analysis.............................................................................................70
Table 9: The comparison results of MILP, GA, and BCO.......................................................71
LIST OF FIGURES

Figure 1: US energy imports ................................................................. 1
Figure 2: The geographical layout .......................................................... 17
Figure 3: Genetic algorithm flowchart ..................................................... 29
Figure 4: Binary encoding ....................................................................... 30
Figure 5: Permutation encoding ............................................................... 31
Figure 6: Value encoding ........................................................................ 31
Figure 7: Tree encoding .......................................................................... 32
Figure 8: Single point crossover ............................................................... 33
Figure 9: Two point crossover .................................................................. 33
Figure 10: Multi-point crossover .............................................................. 34
Figure 11: Simple mutation ...................................................................... 34
Figure 12: Order-based mutation ............................................................. 34
Figure 13: Rank selection ........................................................................ 36
Figure 14: Roulette wheel selection ......................................................... 36
Figure 15: Tournament selection .............................................................. 37
Figure 16: Bee waggle and round dance .................................................... 43
Figure 17: Example of partial solutions after the first forward pass, NC=3, N=3. 44
Figure 18: The first backward pass, NC=3, N=3 ....................................... 46
Figure 19: Dividing bees into two groups, N=3 ........................................ 46
Figure 20: Recruiting of uncommitted followers and the second forward pass, NC=3, N=3 ...... 47
Figure 21: GA schematic diagram for logistic system design problem ........... 52
Figure 22: BCO schematic diagram for logistic system design problem .......... 56
Figure 23: Path development .................................................................. 59
Figure 24: GA annual costs. ...........................................................................................................65
Figure 25: GA percentage of residue that is recirculated per month ........................................66
Figure 26: BCO annual costs. .........................................................................................................68
Figure 27: BCO percentage of residue that is recirculated per month .......................................69
CHAPTER 1: INTRODUCTION

1.1 Background and Research Motivation

Energy produced from renewable sources is receiving more attention than ever before. This is because sources that come from renewable sources can help to diminish the environmental damage caused by burning fossil fuels. The combustion and consumption of fossil fuels such as crude oil, natural gas, and coal causes different greenhouse gases that are released to the atmosphere damaging the health of animals and humans (Haines, Kovats, Campbell-Lendrum, & Corvalan, 2006). According to the Energy Information Agency (EIA), the total energy imports into the US during the years 1973 to 2012 belongs to 601,844 Btu (British thermal units). Only 2% represents the total amount of coal imported during the same period of time. The quantity of natural gas and petroleum products relates to 12% and 19%, respectively, of the total. However, crude oil by itself has a percentage of 67%, which corresponds to a bigger portion of fossil fuels imported to the country Figure1 (U.S. Energy Information Administration, 2013). The dependency of petroleum in the country is a problem, because it is necessary to rely on other foreign countries in order to satisfy the nation’s energy demand. Furthermore, the petroleum supply will eventually run out; it is not infinite. It is estimated that the planet still has 1,333 billion barrels of petroleum to pump out and consume. This is estimated to be enough to cover for approximately 40 years at current usage (BP, 2010).

![US Energy Imports 1973-2012](image)

**Figure 1: US Energy Imports**
Consequently, the focus on bioenergy has become an important alternative to lessen the dependence on petroleum energy. Bioenergy is a renewable energy source produced from biomass, which can be obtained from organic material, waste material, and from wastewater treatment facilities (United States Department of Agriculture, 2009-2014) (Bitman, Mohamed, & Talbi, 2010) (United States Department of Agriculture, 2009-2014) (United States Department of Agriculture, 2009-2014). The importance of biomass is believed to be the only renewable energy source that can offer a reliable substitute for petroleum as stated by EESI (Environmental and Energy Study Institute, 2012).

There are various benefits that can be obtained by developing domestic biomass. One benefit is that the development of biofuels from biomass may help stimulate the economy. In July 2008, the United States spent a total of $74,864,156,500 in imported oil (Otto, 2009). Nevertheless, if biomass is produced in the country, the money spent buying foreign oil will recirculate in our own economy, giving new employment opportunities. Also, it will help mitigate the climate impacts because biofuels offer a reduction in greenhouse gas (GHG) emissions contrary to the petroleum-based fuels. Another benefit is that the country is capable of producing enough biomass to support more than 30% of the internal petroleum consumption (Nerurkar, 2011). The biomass feedstock includes agricultural residue, forest resources, perennial grasses, woody energy crops, algae, and municipal solid waste. Moreover, the dependency that the U.S. has on other foreign countries to obtain fossil fuels will decrease (Office of Energy Efficiency and Renewable Energy, 2012).

On December 19, 2007, a bill was signed in order to create an incentive to increase the amount of renewable biofuels in the United States. It stipulates the use of biomass derived from lignocellulosic biomass and excludes the use of biomass from some other sources. Lignocellulosic biomass produces what is known as second generation biofuels. This type of biomass includes the following: cereal straw, bagasse, forest and crop residues, and purpose grown energy crops (perennial grasses, trees, switchgrass, miscanthus, and crop residues (stems and leaves)) such as vegetative grasses and short rotation forests.
Lignocellulosic presents an advantage over the first generation of biofuels (produced mainly from oil seeds, grains, or sugar beet) since it can be produced faster and at a lower cost. Furthermore, lignocellulosic biomass is an important component of the food crops; this is because it is the part of the plant that is not edible. This means that this type of feedstock does not interfere with food security (Naik, Goud, Rout, & Dalai, 2009).

The bill, named Energy Independence and Security Act (EISA), contains certain provisions about energy efficiency and the availability of use of renewable energy. The setting of the revised Renewable Fuels Standard (RFS) is an important key provision of EISA. The RFS dictates the use of 36 billion gallons per year (BGY) of renewable fuels by 2022. It also specifies fuel allocations for future predictions in 2022. The fuel allocations include the following: 16 BGY of cellulosic biofuels, 14 BGY of advanced biofuels, 1 BGY of biomass-based biodiesel, and 15 BGY of conventional biofuels (Boundy, Diegel, Wright, & Davis, 2011).

Furthermore, the U.S. Department of Energy (DOE) agency aspires to replace 30% of the gasoline used in 2004 with biofuels and produce a total amount of 60 billion gallons per year by 2030. The plan is to obtain 15 billion gallons that come from grains and 45 billion gallons from lignocellulosic biomass (Hess, Kenney, Ovand, Searcy, & Wright, 2009).

In spite of having government support, the amount of biofuels produced from lignocellulosic biomass is advancing slowly. Even though the country has sufficient resources to meet the production requirements, much of those resources are inaccessible using the present biomass supply systems because they are too expensive. That is why the current biomass supply systems are not capable of meeting long term biomass use goals. Consequently, the increasing demand of lignocellulosic biomass creates new logistical challenges to provide an economical, efficient, and reliable supply of feedstock to the biorefineries (Hess, Kenney, Ovand, Searcy, & Wright, 2009).
It is a fact that biomass can make an important contribution in order to supply future energy demands in a sustainable way. However, the supply of sustainable energy is one of the main challenges that men will face. For instance, entrepreneurs and regional planners are facing many logistical challenges to provide an efficient and reliable supply of quality feedstock to biorefineries.

Within the supply chain there may be present different challenges, such as:

- **Seasonal Availability:** Biomass accessibility varies depending on the season and type of the year.

- **Need of Resources:** The limited time frame for collecting large amounts of biomass requires extra-use of equipment and workforce.

- **Low-Density Materials:** The low density of the biomass feedstock increases the need for transportation, handling equipment, and storage space, which can cause a cost increment.

- **Customized Equipment:** Several biomass types need customized collection and handling equipment, which can lead to a complicated structure in the supply chain (Rentizelas, Tolis, & Tatsiopoulos, 2007).

This work is motivated by the urgent need of advancing knowledge and understanding of the complexity of the biofuel supply chain. It is vital to comprehend the logistics of biomass and all the challenges involved in order to design an optimal logistic system that will keep costs low while producing biofuels and also to accelerate the transition from fossil fuels to biofuels.

### 1.2 Thesis Objective

The objective of this thesis is to demonstrate the application of two types of metaheuristic approaches in order to solve a logistic system design problem having three different types of biomass feedstock. This work compares the performance and solutions obtained by two types of metaheuristic approaches: genetic algorithm and bee colony optimization.
Both algorithms are selected because of their effectiveness while solving hard combinatorial optimization problems like the one presented here. It is known that there are much more metaheuristic methods that can also solve combinatorial optimization problems; however, genetic algorithms can be much easier being applied once the problem’s variables are identified. After getting the problem results with the genetic algorithm, the solutions are compared with bee colony optimization. The reason is because bee colony optimization is a relatively new technique developed in the 2000s and was used to compare it with a more traditional methodology such as the genetic algorithm.

Problems with similar characteristics to the one presented in this thesis have been previously solved with the help of complete algorithms, such as linear programming, mixed integer linear programming, and mixed integer programming methods. Mathematical or complete methods are the ones that can provide one with an optimal solution of the problem; however, they may require an exponential computation time to provide a solution of the problem. Approximate algorithms do not guarantee obtaining an optimal solution; instead, they provide feasible solutions but with a minimum amount of computation time.

1.3 Scope and Limitations

The study focuses in a logistic system design problem that was previously solved with a mixed integer linear programming methodology. It analyzes the biomass feedstock logistics since the harvesting process, transportation to intermediate warehouses or biorefineries, biofuels production at the biorefineries, and residue recirculated to the fields. The studies conducted in this thesis use the data collected from Zhu & Yao, 2011. The work aims to maximize the profit of the biofuels production. It considers seven different costs, such as purchasing cost, processing cost, operation cost of biorefineries and intermediate warehouses, transportation cost by trucks and train, harvest units cost, and inventory cost.
As a limitation this work includes the use of only two metaheuristics: GA and BCO. This is considered as a limitation because only those methods were applied and compared. However, this thesis may inspire the application of other methodologies.

1.4 Thesis Outline

This thesis is organized as follows:

Chapter 2 includes literature review about several optimization models (mathematical optimization, linear programming, mixed integer linear programming, and metaheuristic algorithms) that have been used to solve different problems related to the biomass logistic system design area.

Chapter 3 presents the problem description under investigation likewise the data that will be used in the numerical example included in Chapter 5. Furthermore, it explains some of the assumptions considered in the problem. Also, it shows the model and equations necessary for the development of the investigation.

Chapter 4 describes the two metaheuristics used in the thesis investigation: Genetic Algorithm (GA) and Bee Colony Optimization (BCO). It also explains the GAs operators, such as crossover, mutation, and selection methods. Furthermore, an analysis about the GAs encoding type is also presented. Another section talks about BCO as an example on the bees’ behavior in nature and the artificial behavior in the BCO algorithm. It also presents certain applications of the BCO.

Chapter 5 indicates how the logistic system design problem is applied to the GA and BCO. It also includes schematic diagrams that may help to better understand the problem flow. Furthermore, the methodology used in the problem is explained step by step. Moreover, the parameter values, established at the beginning of the problem, are indicated.

Chapter 6 presents the results obtained from the numerical example about the logistic system design problem. Furthermore, the computation results of the problem are displayed and explained in this chapter. A comparison between the mixed integer linear programming, genetic algorithm, and bee
colony optimization is presented. Finally, Chapter 6 presents a description of the sensitivity analysis realized in order to test different parameter values for the two metaheuristic is shown.

Chapter 7 contains general conclusions about the work realized in this thesis besides of important contributions obtained after the elaboration of this work. It also proposes certain investigations that can be done in order to test the performance with other metaheuristic techniques. Furthermore, it proposes to include multiple objectives and to convert the analyzed problem into a more dynamic and robust problem.
CHAPTER 2: LITERATURE REVIEW

This chapter will present an overview of existing logistic problems solved with optimization methods, such as linear programming, mixed integer linear programming, and metaheuristic methods is given.

2.1 Optimization Methods

There are diverse challenges in the design area of a Biomass logistic system causing expensive implementation costs to entrepreneurs. At the present time, different studies have been developed in order to reduce the effect obstacles (such as seasonal availability, need of resources, low-density materials, and customized equipment) have in the implementation costs. The analysis of those studies was possible with the help of some important optimization methods. Many of those optimization problems consist in the search of the best configuration of a set of variables in order to determine a specific goal, or a set of goals. The optimization problems can be classified as those where the solution provides real value results or those where the solutions are encoded with discrete variables (Reche-Lopez, Ruiz-Reyes, Garcia Galan, & Jurado, 2008).

2.1.1 Linear Programming Methods

Problems solved with a linear programming approach, as its name implies, are conformed by linear functions. In linear programming methods, the objective function is also linear, and the constraints are linear equalities or linear inequalities in the unknowns (Luenberger & Ye, 2008). This methodology can be applied to several problems presented in real life.

The next application to be explained is concerning the biomass logistic design where a linear programming approach was developed as a planning tool to calculate the costs generated by transferring the biomass feedstock (switchgrass) from the producers to the biorefinery. The main objective of the study was to minimize the transportation and capacity expansions costs. The analysis took into
consideration the reduction yields due to the storage and handling losses likewise, the production uncertainty due to seasonality (Cundiff, Dias, & Sherali, 2007).

2.1.2 Mixed Integer Linear Programming Methods

Mixed integer linear programming (MILP) theory and its practice has been significantly developed during the years, and MILP is an indispensable tool in engineering and the business area. MILP are very flexible to adapt and can also incorporate many advanced techniques. As a characteristic, mixed integer sets can require both integer constrains and continuous variables. Continuing, there is a brief explanation about different problems that have been solved with the MILP method in some of the biomass logistic challenges that are present today.

In the MILP area, there are various studies developed; for example, in the development of the “Biofeed model” that simulates different feedstock production operations, such as harvesting, packing, storage, handling and transportation, which have an objective to determine the optimal system configuration according to the region. The decision variables of the model include design, planning, and management-level decisions. The model was applied to a case study for the State of Illinois where switchgrass was produced; the results showed that the price varied depending on the collection region. It also provided the biorefineries capacity and the average utilization of the transportation fleet. It also revealed that the lack of in-field and centralized storage reduced the system profit between 5% to 17%. At the end, the optimization of the harvesting schedule increased the total profit to 30% (Shastri, Hansen, Rodriguez, & Ting, 2011).

A MILP model was formulated in order to maximize the profit of biofuel that is produced from lignocellulosic biomass; the study deals with a multi-commodity system, production, distribution, identifying facility locations, capacities, technologies, and material flow (An, Wilhelm, & Searcy, 2011).
Furthermore, another study proposes a MILP model, which composes the supply chain and logistics decisions and has as objective to minimize the total annualized switchgrass costs based on bioethanol supply-chain costs. The application of the model demonstrated that by using only 61% of the available land for the switchgrass production, 100% of the gasoline required in the North Dakota state can be met with the production of bioethanol (Zhang, Osmani, Awudu, & Gonela, 2012).

For instance, the design and management of the biomass to the biorefinery has inspired the proposal of different mixed integer programming methods to alleviate the supply-chain issues. One study was used to design the supply chain and manage the logistics of the biorefinery. The proposed model coordinates the supply-chain decisions and determines the number, size, location of biorefineries, the amount of biomass shipped, processed, and inventory levels. The two sources of biomass feedstock considered in this study are corn stover and woody biomass (Eksioglu, Acharya, Lieghtlay, & Arora, 2009).

Another problem formulated with MILP enables the decision making for the selection of biofuel conversion technologies including locations, volumes, supply networks, and the transportation logistics from forestry resources to the final market. Besides that, its objective is to maximize the function of overall profit. The model includes very basic information about transportation, capital, and operating costs. Also, there are limitations about considerations on conversion technologies, and the model does not take into account different factors within the existing wood processing infrastructure (Kim, Realff, Lee, Whittaker, & Furtner, 2011).

Also, this methodology has been used to model the optimization of the integrated biofuels supply chain. It takes into account the technical and economical parameters that may affect the performance of the whole supply chain. The investigation takes into account some biofuel problems because this area is constantly changing in the economic and strategic decisions about what land should be dedicated to biofuels, the production of biofuels in the country or imports, and the domestic consumption or exports.
This approach has the ability to identify and solve problems about facility location, raw material selection, trading policy, design, selling prices, or quantities to be imported or exported (Papapostolou, Kondili, & Kaldellis, 2011).

Analyzes about intermodal facilities (facilities where two or more transportation modes meet: rail ramps, land and sea ports) has been done in order to design and manage the supply chain of the biorefinery. A MILP approach was utilized to identify the impact of these types of facilities to design a proper management of the total infrastructure. The available data of the model includes location of the facilities, transportation modes, cost, and cargo capacities for each transportation type, the geographical distribution of the biomass feedstock (logging residuals, thinning, prunings, grasses, and chips/shavings), production yields, biomass processing, and inventory costs. The results from this model are the location, capacity, and number of production plants. It also determines for each plant the transportation mode, shipments timing and size, inventory size and production schedule that minimizes the delivery costs of biofuel (Eksioglu, Analyzing the Impact of Intermodal Facilities to the Design and Management of Biofuels Supply Chain, 2009).

A different problem examines the challenges in the designing of the logistic system for the biomass industry. Those challenges are due to the low bulk density, restrictions of the harvesting season, weather effects, and distribution over a certain geographical area of the switchgrass biomass feedstock. Under this condition, a MILP model was created. The part of the supply chain, studied in this scenario, covers the planting of switchgrass, harvesting, likewise the delivery of the biomass to the biorefinery, and residue handling. Its main concentration provides a decision support aid in order to design the supply chain of the system under study and year round tactical schedules that varied due to the switchgrass yields. At the end, the deterministic optimization method provided a pathway for the production of the feedstock to the biorefinery (Zhu, Li, Yao, & Chen, 2011).
One more research established the MILP logistic model for the biomass industry with different types of biomass feedstock such as switchgrass, corn stalk, and wheat straw. This project demonstrates that it is possible to integrate a variety of biomass feedstock such as the perennial grasses and agricultural residues. The model determines the location of warehouses, the size of the harvesting team, the types and amounts of biomass harvested/purchased, stored and processed per month, and the transportation type. It also shows the advantages of using multiple biomass feedstocks by doing a comparison of using only a single biomass feedstock (switchgrass) (Zhu & Yao, 2011).

2.1.3 Metaheuristic Methods

There are other techniques that can be used in order to solve very complex problems such as the combinatorial optimization problems (CO). Some important examples of combinatorial optimization problems involve the Traveling Salesman Problem (TSP), the Quadratic Assignment problem (QAP), Timetabling, and Scheduling problems. Due to the complexity of the combinatorial optimization problems, different algorithms have been developed. They can be classified as either complete or approximate algorithms. Complete algorithms are the ones that can guarantee to find an optimal solution at the expense of high computation times for practical purposes. Thus, the approximate methods to solve CO problems are receiving increasingly attention nowadays. In the approximation methods, the guarantee of finding optimal solutions is sacrificed for the sake of getting good solutions in a significantly reduce amount of time.

In the last 20 years, a new kind of approximate algorithms has emerged. The described algorithm has the capacity of guiding a subordinate heuristic for exploring and exploiting a determined search space to find efficiently near-optimal solutions. This class of algorithms is called metaheuristic. Some of the most common metaheuristics are: Ant Colony Optimization (ACO), Evolutionary Computation (EC), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Bee Colony Optimization (BCO), Iterated Local Search (ILS), Simulating Annealing (SA), and Tabu Search (TS).
There are several approaches to classified metaheuristics, according to their properties. The most important ways of classifying metaheuristics are (Blaum & Roli):

- Nature-inspired: Some examples are GA, ACO, and BCO.
- Non-nature-inspired: Includes TS and ILS.
- Population-based: Perform search processes that describe the evolution of a set of points in the search space (GA, BCO, and PSO).
- Trajectory methods: Algorithms that work on a single solution at any time and encompass single-point search-based metaheuristics (TS and ILS).

In the next chapter, there is a brief description about some metaheuristic algorithms applied in the research of particular areas concerning biomass logistic challenges.

For example, a study where four important metaheuristics were used and compared to determine the optimal location and supply area of biomass-fueled power plants was performed. Two trajectory methods such as the SA and TS, and two population-based methods like GA and PSO were considered for the particular problem. In the investigation, a new binary PSO was proposed and applied to the problem. As a fitness function, the profitability index of the biomass power plant was used. The plant technology is based on gas turbines to produce electric energy from forest residues.

The thesis experimental results show that the proposed algorithm (BPSO) provides better solutions than GA and other proposed BPSO algorithms. Meaningful results about the proper coordinates of the optimal location, supply area, profitability index, net present value, initial investment, distance to grid, the generated power of the plant are obtained for each type (Reche-Lopez, Ruiz-Reyes, Garcia Galan, & Jurado, 2008).

Another investigation related to the determination of the facility optimal location, finding the biomass supply area and power plant size which could offer the best profitability to the investors, was developed. However, the focus of this study was to calculate a new tool based on PSO (BHBF- Binary
Honey Bee Foraging algorithm). This approach efficiently determined the optimal location of the biomass power plant avoiding an exhaustive computation search. The biomass feedstock analyzed is olive trees that are cultivated worldwide, especially in Mediterranean countries. More than 97% are produced in that region (Spain, Italy, and Greece) (Vera, Carabias, Jurado, & Ruiz-Reyes, 2010).

2.2 Summary

This chapter presented different optimization methods, such as mathematical optimization, linear programming, mixed integer linear programming, and metaheuristic models that have been applied in the investigation of different studies on the biomass logistic system design. Also, some of the literature analyzes the entire supply chain of a particular system or focuses on a certain area within the supply chain. A similarity in all the research mentioned in the literature review include as a primary objective at the end to minimize costs or the maximization of profit.
CHAPTER 3: LOGISTIC SYSTEM DESIGN PROBLEM

The implementation of multi-commodity systems can mean a great advancement in the economical aspect of the entire biomass logistic system. Thus, it is very important to take into consideration the several advantages that this type of system can offer. Such as in the total system cost reduction, saving in the stage of storage and additional cost savings could be expected from smoother resource requirements at the biomass supply chain, including equipment and labor (Rentizelas, Tolis, & Tatsiopoulos, 2007). However, there are other important challenges within this type of system that prevent entrepreneurs to invest in the design of these types of systems. Consequently, these types of issues are not permitting to achieve the goals established in the Energy Independence and Security Act in 2007 in order to allow the increment of the biofuel production in the country.

3.1 Problem Description

Several of the limitations and challenges while designing this type of system include the transportation network, feedstock supply, residue handling, and the tactical operation schedules. One of the studies done about multi-commodity systems proposed a MILP model that includes a variety of biomass feedstock, including native perennial grasses and agricultural residues such as switchgrass and corn stalk & wheat straw respectively. It demonstrated the advantages of using different biomass feedstocks with the help of some numerical examples, and the model takes into account the biomass characteristics, such as seasonality, residue recirculated to the fields, strategy decisions; for instance, distribution via direct transportation or transshipment through intermediate warehouses. Additionally, tactical schedules, for example, monthly operations: types, purchased, stored, processed, and amounts of biomass harvested (Zhu et. al 2011).

This problem motivated the investigation of this research, due to, at the present time, the biomass to bioenergy industry is facing critical issues regarding the use of a single biomass feedstock because that raw material is not sufficiently available and reliable to operate a plant economically.
speaking. However, it is known that there are multiple types of biomass available in a certain region (Bracmort, 2010). That is why, based on the results obtained in the previous study, it was decided to solve the same problem with two different techniques: GA and BCO (Epplin, Clark, Roverts, & Hwang, 2007). It was done mainly to demonstrate that the metaheuristic techniques GA and BCO can be adapted to problems like this one and show the superior advantages that these models have over the mathematical model developed in the previous study. Details of GA and BCO will be discussed in Chapter 4. Furthermore, according to the literature review done in this area, it is valid to say that the present investigation is the first one that addresses the issue of multiple types of biomass feedstock in the logistic system design perspective using the mentioned metaheuristic algorithms.

3.2 Problem Assumptions

The presented model in this thesis can be adapted to address a problem including different types of biomass feedstocks, but in this particular work, the study focuses only on three types of biomass: switchgrass, corn stalk, and wheat straw. They are used as a feedstock in order to produce biofuel. However, it is important to specify the different assumptions that were considered in the research. These assumptions describe the biomass characteristics and the multiple feedstock system. Other assumptions may be clarified in a specific time through the elaboration of this work.

Assumptions:

1. Switchgrass cannot be harvested during the months of March to June (non-harvesting season) (Mapemba, Epplin, Taliaferro, & Huhnke, 2007).

2. Due to the seasonality of the switchgrass; it is assumed that corn stalk and wheat straw can be obtained during the months of March to June (Epplin, Mapenba, & Tembo, 2005).

3. The central management of the system is only in charge of the switchgrass harvesting due to the corn stalk and wheat straw are purchased outside the system.

4. The biomass residue, generated in the biorefinery, is recirculated to the switchgrass fields.
5. There are two transportation modes: trucks and trains. The trains are available for the locations near to the rails.

6. The transported biomass feedstock is assumed to be already dry (dry tons is used as a unit to measure the content of dry materials)

3.3 Data

The studies conducted in this thesis use the data collected from Zhu & Yao, 2011. The analyze problem involves the use of two biorefineries, three intermediate warehouses, ten switchgrass fields, two corn stalk fields, and two wheat straw fields. The study was conducted in the theoretical geographical layout shown in Figure 2. Where $F_{1\to 10}$ represents the number of switchgrass production fields, $S_{11\to 12}$ are fields where corn stalk is produced, $T_{13\to 14}$ indicates the fields from which the wheat straw is harvested, $W_{15\to 17}$ are the three intermediate warehouses, $B_{18\to 19}$ are the two biorefineries where the biofuel is produced.

![Figure 2: The geographical layout](image)

The transportation types considered in the problem as previous stated are trucks and trains. All the facilities are accessible by trucks, but biorefinery 18, corn stalk field 14, switchgrass field 8, intermediate warehouse 16, and biorefinery 19 are also reachable by train. The production capacity ($BCAP_{km}$) of each biorefinery is 120,000 dry tons/month. The total yield of the three types of biomass is
2,100,000 dry tons/year. The fixed cost for operating an intermediate and in-biorefinery warehouse \( (\mu_j) \) is $60,000 and $30,000 per month. However, the in-field warehouses do not incur in any operating costs. The cost of storing any biomass feedstock \( (\alpha_{ijm}) \) and residue \( (\alpha_{0jm}) \) is $2/dry ton/month in any of the potential warehouses. The storage capacity \( (SCAP_j) \) is 200,000 dry tons/month for all intermediate warehouses, \( (W_{15}, W_{16}, W_{17}) \), 60,000 dry tons/month for in-biorefinery warehouses, and unlimited for in-field warehouses.

The cost for operating a biorefinery is \( \nu_k = $10,000,000 \) per year. The sale price of the biofuel is \( \rho_m = $1.8 \) /gallon. This price was calculated according to the current potential market price. The processing cost for all three feedstocks is \( c_{mkl} = $50/ \) dry ton. The purchase price for stalk and straw is the same \( m_2 = m_3 = $35/ \) dry ton and for switchgrass it is \( m_1 = $50/ \) dry ton. The conversion equivalence for all three types of biomass is a dry ton of biomass that can be converted into 90 gallons of biofuel and 0.01 tons of residues. The factor \( \psi = 0.005 \) indicates a specific proportion of residue that must be re-circulated to the switchgrass fields per year. The residue is re-circulated to the fields because it serves as a fertilizer for the preservation of minerals and quality of the soil.

The harvest unit capacity is 7200 dry tons/month, and its annual operating and maintenance cost is \( \gamma = $580,000 \). The harvest units are used specifically for the harvesting of switchgrass because it is the only feedstock where the central management of the system is in charge. A harvesting team includes: 10 workers, 9 tractors, 3 mowers, 3 balers, 3 rakes, and 1 field transporter.

The transportation capacity is unlimited for trucks and trains. The transportation cost between two destinations is the same for biomass feedstock and residue. Train transportation has a unit cost of $0.04 per mile/dry ton and truck transportation unit cost is $0.40 per mile/dry ton. The factors \( \xi = 0.2 \) (value > 0) and \( \delta = 0.5 \) (value < 1) define the minimum processed biomass feedstocks, and the second factor indicates the minimum biomass inventory in biorefineries.
3.4 Model

This section presents the MILP for the multi-commodity network flow for the biomass-to-bioenergy logistics system design. The problem involves a single objective function, which is profit maximization. To be able to calculate profit, first of all, it is necessary to calculate the revenue of the system in a determined period of time. In this problem, the model is suitable for calculations per year or month. After knowing the revenue of the system, it is also required to identify the total costs of the multi-commodity system. Then, the profit can be obtained using equation one.

\[
\text{max } R - \sum_{n=1}^{7} C_n
\]  

(1)

Where:

\( R \) = Revenue

\[ \sum_{n=1}^{7} C_n \] = Represent the total annual costs considered in the system.

Equation two calculates the revenue generated per month; that calculation is possible after knowing the total amount of gallons of biofuel produced in both biorefineries. The quantity of gallons produced is generated with the use of the three different biomass feedstocks discussed in the thesis (switchgrass, corn stalk, and wheat straw).

\[
R = \sum_{m=1}^{12} \sum_{k=1}^{2} \sum_{l=1}^{3} \rho_m b_{mkll}
\]  

(2)

Where:

\( \rho_m \) = Price of a gallon of biofuel in month \( m \).

\( b_{mkll} \) = Total gallons of biofuel produced.
The following equations indicate specifically what types of costs are taken into consideration in the system under study:

- **Processing cost**

\[ C_1 = \sum_{m=1}^{12} \sum_{k=1}^{2} \sum_{l=1}^{3} C_{mkl} P_{mkl} \]  

(3)

Where:

- \( P_{mkl} \) = Total amount of dry tons of biomass \((l)\) processed at biorefinery \((k)\) during month \((m)\).

- \( C_{mkl} \) = Processing cost of a dry ton of biomass \((l)\) processed at biorefinery \((k)\) during month \((m)\).

- **Feedstock purchasing cost**

\[ C_2 = \sum_{m=1}^{12} \left[ m_1 \sum_{i=1}^{10} h_{kim1} + m_2 \left( \sum_{i=1}^{2} \sum_{j=1}^{3} t_{skm1} \right) + m_3 \left( \sum_{i=1}^{2} \sum_{j=1}^{2} t_{tkm1} \right) + m_4 \left( \sum_{i=1}^{2} \sum_{j=1}^{2} t_{tkm2} \right) \right] \]  

(4)

Where:

- \( m_{1,2,3} \) = Purchasing costs of switchgrass (1), stalk (2), and straw (3).

- \( h_{kim1} \) = Dry tons of switchgrass harvested from field \((i)\) during month \((m)\).

- \( t_{skm1} \) = Dry tons of biomass transported by truck to biorefinery \((k)\) from field \((s)\) during month \((m)\).

- \( t_{skm2} \) = Dry tons of biomass transported by train to biorefinery \((k)\) from field \((s)\) during month \((m)\).

- \( t_{tkm1} \) = Dry tons of biomass transported by truck to biorefinery \((k)\) from field \((t)\) during month \((m)\).

- \( t_{tkm2} \) = Dry tons of biomass transported by train to biorefinery \((k)\) from field \((t)\) during month \((m)\).
• Inventory cost of biomass and residue

\[ C_3 = \sum_{m=1}^{12} \sum_{j \in J} \left( \sum_{i=1}^{3} \alpha_{ijm} s_{ijm} + \alpha_{0jm} s_{0jm} \right) \]  

Where:

- \( \alpha_{ijm} \) = Inventory cost of biomass feedstock \( I \) keep it in warehouse \( J \) during month \( M \).
- \( \alpha_{0jm} \) = Inventory cost of residue \( 0 \) keep it in warehouse \( J \) during month \( M \).
- \( s_{ijm} \) = Dry tons of biomass \( I \) stored at warehouse \( J \) during month \( M \).
- \( s_{0jm} \) = Dry tons of residue stored at warehouse \( J \) during month \( M \).

• Transportation cost by truck

\[ C_4 = \sum_{m=1}^{12} \left\{ \sum_{i=1}^{10} \sum_{j \in J} \beta_{ijm11} t_{ijm11} + \sum_{i=1}^{10} \sum_{k \in K} \beta_{ikm11} t_{ikm11} + \sum_{k=1}^{10} \sum_{j \in J} \beta_{kmj11} t_{kmj11} + \sum_{i=1}^{10} \sum_{k \in K} \beta_{kim11} t_{kim11} \right\} \]  

Where:

- \( \beta_{ijm11} \) = Transportation cost by truck \( 1 \) of switchgrass \( 1 \) from switchgrass field \( I \) to warehouse \( J \) during month \( M \).
- \( t_{ijm11} \) = Transportation by truck \( 1 \) of switchgrass \( 1 \) from switchgrass field \( I \) to warehouse \( J \) during month \( M \).
- \( \beta_{jkmm11} \) = Transportation cost by truck \( 1 \) of switchgrass \( 1 \) from warehouse \( J \) to biorefinery \( K \) during month \( M \).
- \( t_{jkmm11} \) = Transportation by truck \( 1 \) of switchgrass \( 1 \) from warehouse \( J \) to biorefinery \( K \) during month \( M \).
\( \beta_{ikm12} \) = Transportation cost by truck (1) of switchgrass (1) from switchgrass field \((i)\) to biorefinery \((k)\) during month \((m)\).

\( t_{ikm11} \) = Transportation by truck (1) of switchgrass (1) from switchgrass field \((i)\) to biorefinery \((k)\) during month \((m)\).

\( \beta_{skm21} \) = Transportation cost by truck (1) of stalk (2) to biorefinery \((k)\) during month \((m)\).

\( t_{skm21} \) = Transportation by truck (1) of stalk (2) to biorefinery \((k)\) during month \((m)\).

\( \beta_{tkm31} \) = Transportation cost by truck (1) of straw (3) to biorefinery \((k)\) during month \((m)\).

\( t_{tkm31} \) = Transportation by truck (1) of straw (3) to biorefinery \((k)\) during month \((m)\).

\( \beta_{kim01} \) = Transportation cost by truck (1) of residue (0) from biorefinery \((k)\) to switchgrass field \((i)\) during month \((m)\).

\( t_{kim01} \) = Transportation by truck (1) of residue (0) from biorefinery \((k)\) to switchgrass field \((i)\) during month \((m)\).

- Transportation cost by train

\[
C_5 = \sum_{i=1}^{12} \left( \frac{\sum_{j=1}^{10} \sum_{k=1}^{12} \beta_{jkm12} + \sum_{j=1}^{12} \sum_{k=1}^{10} \beta_{jmk12} + \sum_{j=1}^{12} \sum_{k=1}^{10} \sum_{m=1}^{12} \beta_{jm21} + \sum_{j=1}^{10} \sum_{k=1}^{12} \sum_{m=1}^{12} \beta_{jkm12}}{12} \right)
\]

\( \beta_{ijm12} \) = Transportation cost by train (1) of switchgrass (1) from switchgrass field \((i)\) to warehouse \((j)\) during month \((m)\).

\( t_{ijm12} \) = Transportation by train (1) of switchgrass (1) from switchgrass field \((i)\) to warehouse \((j)\) during month \((m)\).

\( \beta_{jkm12} \) = Transportation cost by train (1) of switchgrass (1) from warehouse \((j)\) to biorefinery \((k)\) during month \((m)\).
\( t_{jkm12} \) = Transportation by train (1) of switchgrass (1) from warehouse \((j)\) to biorefinery \((k)\) during month \((m)\).

\( \beta_{tkm12} \) = Transportation cost by train (1) of switchgrass (1) from switchgrass field \((i)\) to biorefinery \((k)\) during month \((m)\).

\( t_{tkm12} \) = Transportation by train (1) of switchgrass (1) from switchgrass field \((i)\) to biorefinery \((k)\) during month \((m)\).

\( \beta_{skm22} \) = Transportation cost by train (1) of stalk (2) to biorefinery \((k)\) during month \((m)\).

\( t_{skm22} \) = Transportation by train (1) of stalk (2) to biorefinery \((k)\) during month \((m)\).

\( \beta_{tkm32} \) = Transportation cost by train (1) of straw (3) to biorefinery \((k)\) during month \((m)\).

\( t_{tkm32} \) = Transportation by train (1) of straw (3) to biorefinery \((k)\) during month \((m)\).

\( \beta_{kim02} \) = Transportation cost by train (1) of residue (0) from biorefinery \((k)\) to switchgrass field \((i)\) during month \((m)\).

\( t_{kim02} \) = Transportation by train (1) of residue (0) from biorefinery \((k)\) to switchgrass field \((i)\) during month \((m)\).

- Operation cost of warehouses and biorefineries

\[
C_6 = \sum_{m=1}^{12} \sum_{j \in J} \mu_j y_{jm} + \sum_{k=1}^{2} v_k \]

Where:

\( \mu_j \) = Fixed operational costs when using an intermediate warehouse and in-biorefinery warehouse.

\( y_{jm} \) = Binary value (0 or 1); if is equal to one means that warehouse \((j)\) is open during month \(m\) and (0) otherwise.

\( v_k \) = Operation cost of biorefinery \((k)\).
\( z_k \) = Binary value (0 or 1); if is equal to one means that biorefinery \((k)\) is open during
month \(m\) and (0) otherwise.

- Operation cost of harvest units

\[
C_\gamma = \gamma u \tag{9}
\]

Where:

\[
\gamma = \text{Annual operating and maintenance cost of the harvest unit.}
\]

\[
u = \text{Number of employed harvest units.}
\]

Following there are the restrictions found in the problem. It is very important to consider them in the
model while calculating the optimal value of the objective function:

- Production capacity

\[
\sum_{l=1}^{3} p_{mkl} \leq BCAP_{km} z_k \tag{10}
\]

Where:

\[
p_{mkl} = \text{Total amount of dry tons of biomass \((l)\) processed at biorefinery \((k)\) during month \((m)\).}
\]

\[
BCAP_{km} = \text{Production capacity at biorefinery \((k)\) during month \((m)\).}
\]

\[
z_k = \text{Binary value (0 or 1); if is equal to one means that biorefinery \((k)\) is open during
month \((m)\) and (0) otherwise.}
\]

The production capacity constraint indicates that the total amount of dry tons of biomass
processed in any of the two biorefineries must be less or equal to the biorefinery production capacity.
The total dry tons of biomass can include any of the biomass feedstock analyzed in the problem.

- Safety inventory level

\[
\sum_{l=1}^{3} p_{mkl} \geq \varepsilon BCAP_{km} z_k \tag{11}
\]
Where:

\[ BCAP_{km} = \text{Production capacity at biorefinery (k) during month (m)}. \]

\[ \xi = \text{Factor that define the minimum processed biomass in the biorefineries (value > 0)}. \]

\[ z_k = \text{Binary value (0 or 1); if is equal to one means that biorefinery (k) is open during month (m) and (0) otherwise.} \]

This restriction indicates a safe inventory level with the minimum processed biomass feedstock in each biorefinery to avoid unexpected interruptions of biomass supply and biofuel production.

- Storage capacity

\[ \sum_{j=1}^{3} s_{jm} \geq SCAP_j y_{jm} \quad (12) \]

Where:

\[ s_{jm} = \text{Amount of biomass feedstock (l) storage in warehouses (j) during month (m)}. \]

\[ SCAP_j = \text{Safety storage capacity for intermediate warehouses and in-biorefinery warehouses}. \]

\[ y_{jm} = \text{Binary value (0 or 1); if is equal to one means that warehouse (j) is open during month (m) and (0) otherwise.} \]

The storage capacity constraint on warehouses indicates a limit on the biomass feedstock and residue that can be stored at the different warehouse location. Nevertheless, this limit is only for intermediate and in-biorefinery warehouses because in-field warehouses are assumed to have an infinite capacity.

- Safety inventory level

\[ \sum_{j=1}^{3} s_{tkm} \geq \delta SCAP_k y_{km} \quad (13) \]

Where:

\[ s_{tkm} = \text{Amount of biomass feedstock (l) in inventory at biorefinery (k) during month (m)}. \]

\[ \delta = \text{Factor that defines the biomass inventory in biorefineries (value<1)}. \]
\( SCAP_k \) = Safety inventory capacity at biorefinery \((k)\).

\( y_{km} \) = Binary value (0 or 1); if is equal to one means that the biorefinery \((k)\) has inventory and (0) otherwise.

This constraint imposes a safety inventory level on the minimum stored biomass feedstock in each biorefinery to avoid unexpected interruptions of biomass supply and biofuel production.

### 3.5 Summary

This chapter began by indicating the advantages and limitations while using multi-commodity systems. However, since the biggest logistic challenge in the design of a biomass feedstock system is the huge implementation cost when planning to operate a system like this one. It is important to investigate and do more research about systems that can implement different types of biomass feedstock due to the fact that they can offer an increment in the economical yield of the system. It also demonstrates what motivates this research and the reasons why the biomass logistic design problem was selected. Furthermore, a description of the problem analyzed in the thesis is given. Moreover, an explanation and the data collected about the problem are mentioned. In the last section of the chapter, the mathematical model is presented and explains all the equations and constraints taken into consideration regarding this issue.
CHAPTER 4: METAHEURISTIC ALGORITHMS

Metaheuristics can be seen as high level strategies that allow the exploration of search spaces by using different methods. This thesis is focused on the implementation of two important metaheuristic methods: Genetic algorithms (GA) and Bee Colony optimization (BCO) in order to solve a biomass logistic system design which includes different types of biomass feedstocks. Therefore, a more detailed description about GAs and BCO is presented throughout this chapter.

4.1 Genetic Algorithm

The GA was developed by John Holland during the 1970s. A GA conceptually follows steps inspired by the biological processes of evolution. Evolution is a method that permits searching among different numbers of possibilities or solutions. In biological terms, those solutions are sets of possible genetic sequences, and the best solutions are the organisms that are able to survive and reproduce in their environments. GA is a nondeterministic stochastic optimization method that simulates evolution processes in order to solve very complex problems. In order to start using a GA, first, an initial population of chromosomes (also called individuals) is generated; usually, this is done randomly. During every generation, in GA, each individual of the population is evaluated with the appropriate fitness function according to the problem. Then, some individuals will reproduce new individuals by reproduction. The reproduction process requires crossover and mutation operators. The mutation operator will create new individuals by making a little change in a single individual, and the crossover operator will create new individuals by combining parts of two individuals. The new individuals generated by the reproduction process are called offspring, which are then evaluated. Next, a selection procedure is performed to select the new members of the new population that is formed by parent population and offspring population. That selection process will be based on its fitness value and the individuals with high-fitness value will have better opportunity to reproduce. However, the individuals with a low fitness value are more likely to disappear. After several generations, the procedure is
terminated either when the process evaluates all the search space or when a determined number of iterations is reached. It is very important to mention that GAs are processes that may not guarantee an optimal solution, especially because the termination condition may be stopped in a feasible solution (Mitchell, 1999).

### 4.1.1 Pseudo-code

The pseudo-code of a simple GA works as follows (Mitchell, 1999):

**Step-1:** **Start:** Generate the initial population of $N$ number of chromosomes or individuals.

**Step-2:** **Fitness:** Calculate the fitness $f(x)$ of each chromosome $x$ in the population.

**Step-3:** **New population:** Create the new population until the termination criterion is complete.

**Step-3.1:** **Select:** a pair of chromosomes from the current population so they can reproduce.

**Step-3.2:** **Crossover:** With probability $p_c$ (“crossover probability” or “crossover rate”) cross over the pair at a randomly chosen point in order to create two offspring.

**Step-3.3:** **Mutation:** With probability $p_m$ (“mutation probability” or “mutation rate”) mutate new offspring at each position in the chromosome.

**Step-4:** **Replace:** The current population with the new population of offspring and chromosomes to run the algorithm.

**Step-5:** **Check:** If the stopping criterion is satisfied, stop the algorithm and display the best solution of the population.

**Step-6:** **Go to step 2:** If the stopping criterion is not satisfied.
Figure 3: Genetic algorithm flowchart
4.1.2 Chromosome Implementation

How to encode the solutions of the problem under investigation into chromosomes is an important issue when using genetic algorithms. Various encoding methods have been developed for a determined type of problem in order to offer an effective and efficient implementation of the GAs. The existent encoding methods can be classified as follows:

- Binary encoding
- Permutation encoding
- Value encoding
- Tree encoding

4.1.2.1 Binary Encoding

Binary encoding is the most commonly used; this is because since the beginning of GA, Holland and his students started their investigations with this type of encoding and because binary encoding is very simple to use. In binary encoding, every chromosome is formed by strings of bits, including zeros or ones. Each bit in the chromosome can represent a particular characteristic in the problem, or the whole chromosome may represent a single number (Mitchell, 1999).

<table>
<thead>
<tr>
<th>Chromosome A:</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome B:</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 4: Binary encoding

4.1.2.2 Permutation Encoding

In integer or permutation encoding, every chromosome is formed by a string of numbers which represent a position in a sequence. Sometimes, this type of encoding is best used for combinatorial optimization problems because the idea of these kinds of problems is to find the best permutation or combinations of items which may be subject to certain number of constraints. A good example about
this type of encoding is the traveling salesman problem. This problem can have as an objective to minimize the distance traveled or the costs for passing through a determined path. In this example, the bit values presented in the chromosome can represent six cities that the salesman must visit and the chromosome can be a representation of the paths that the salesman can visit in that specific order (Mitchell, 1999).

<table>
<thead>
<tr>
<th>Chromosome A:</th>
<th>4</th>
<th>1</th>
<th>5</th>
<th>3</th>
<th>6</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome B:</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 5: Permutation encoding

4.1.2.3 Value Encoding

Value encoding is used in problems where real values or specific data is provided. This type of encoding is best used for function optimization problems. It is well known that real value chromosomes perform better than binary encoding for function optimization and problems with restrictions or constraints. The real value encoding can be a sequence of values such as real numbers, objects, or characters (Mitchell, 1999).

<table>
<thead>
<tr>
<th>Chromosome A:</th>
<th>1.55</th>
<th>3.54</th>
<th>5.46</th>
<th>2.87</th>
<th>6.90</th>
<th>2.32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome B:</td>
<td>S</td>
<td>E</td>
<td>W</td>
<td>N</td>
<td>S</td>
<td>W</td>
</tr>
<tr>
<td>Chromosome C:</td>
<td>D</td>
<td>A</td>
<td>C</td>
<td>B</td>
<td>F</td>
<td>A</td>
</tr>
</tbody>
</table>

Figure 6: Value encoding

4.1.2.4 Tree Encoding

Tree encoding schemes have different advantages because they allow the search space to be open-ended. However, they also have certain disadvantages due to the fact that they can grow in uncontrolled forms.
Nevertheless, it can be very useful for evolving programs or for other structures that can be encoded in trees. The tree encoding is formed by objects such as commands or functions in a programming code or language (Mitchell, 1999).

![Tree encoding](image)

**Figure 7: Tree encoding**

### 4.1.3 Genetic Algorithm Operators

The two basic algorithm operators are crossover and mutation; both operators influence the performance of the algorithm. The crossover operator is vital for the reproduction of new chromosomes, and the mutation operator helps to include certain randomness in the search space. The two of them are of great relevance in the algorithm. Following is an overview about the different types of crossover and mutation techniques.

#### 4.1.3.1 Crossover Techniques

A main feature of the GA is the crossover operator because it allows the production of new chromosomes (sons) by the exchanging of information (genes) among two pairs of chromosomes or individuals (parents). The types of crossover techniques are diverse such as the single point, two point, multiple-point, uniform, multi-parent, order, partially mapped, cycle, and linear order crossover (Picek, Golub, & Jakobovic, 2010). However, some of the most common crossover techniques are presented next:
4.1.3.1.1 Single Point Crossover

One common crossover method is the single point crossover. In this technique the position of a single point (cutting point) is selected randomly. After that, the parts of the two parents are exchanged in order to form two offspring (Ferrolho & Crisostomo, Manuel).

![Figure 8: Single point crossover](image)

4.1.3.1.2 Two Point Crossover

In two point crossover, two cutting positions are selected randomly, and the values are exchanged between these cutting points, causing the production of two offspring (Ferrolho & Crisostomo, Manuel).

![Figure 9: Two point crossover](image)

4.1.3.1.3 Multi-Point Crossover

In multi-point crossover, $N$ cutting point positions are selected randomly. After the random selection, the variable values are interchanged to create two offspring. This type of technique encourages the exploration of the search space (Mitchell, 1999).
4.1.3.2 Mutation Techniques

The second basic algorithm operator is mutation because it permits a level of variation and randomness in the algorithm. The different mutation methods are: single point, multi-point, bit inversion, order-based, position-based, and scramble. Following is a list of the most common mutation techniques (Ferrolho & Crisostomo, Manuel).

4.1.3.2.1 Simple Mutation

In simple mutation (single point or bit inversion) a single gene of the chromosome or individual is randomly selected to be mutated, and its value is changed with one value according to the encoding type used (Ferrolho & Crisostomo, Manuel).

<table>
<thead>
<tr>
<th>Chromosome</th>
<th>Mutated chromosome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 0 1 1 1</td>
<td>1 1 0 1 1 1</td>
</tr>
</tbody>
</table>

Figure 11: Simple mutation

4.1.3.2.2 Order-based Mutation

In order-based mutation, two positions are simply chosen and randomly, and their values are exchanged as shown below in Figure 12 (Ferrolho & Crisostomo, Manuel).

<table>
<thead>
<tr>
<th>Chromosome</th>
<th>Mutated chromosome</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 5 2 1 7 6</td>
<td>4 5 7 1 2 6</td>
</tr>
</tbody>
</table>

Figure 12: Order-based mutation
4.1.4 Selection Methods

The selection process in GAs determines how many times a particular individual or chromosome will be selected for reproduction. The selection process also controls the number of times that the individual will be reproduced. When the algorithm starts to work; it is recommended to do a wide exploration of the search space, and the selection is mainly recommended at the end in order to narrow the search space. The reason is because the selection will occur mainly between higher fitness individuals, which will improve the algorithm performance. At the present time, there are different selection methods that can be applied in Gas, but the most common methods are:

- Rank selection
- Roulette wheel selection
- Tournament selection
- Steady state selection
- Elitism

4.1.4.1 Rank Selection

In the ranking selection method, the chromosomes or individuals of the population are sorted from the best to the worst fitnest values. After sorting, each individual in the population will be assigned a ranking number based primarily on its fitness value. However, the selection is based in this ranking number instead on its fitness differences. The main advantage of this selection type is to prevent very fit individuals from getting more dominance or control too early in the process over the less fit ones. If the chromosomes with more fitness value gains control, the population would not be diverse and may have troubles to find an acceptable solution. The only disadvantage of this technique is that it requires ranking all the members of the population which will take a considerable computation time (SivaraJ & Ravichandran Dr., 2011).
4.1.4.2 Roulette Wheel Selection

The roulette selection method is one of the most common GA selection procedures in the determination of the potential chromosomes or individuals that will be used with the crossover and mutation operators. In this technique, like in the other ones, a fitness value is assigned to all the possible solutions generated from the individuals in the population, and as mentioned before, the value is obtained through the fitness function. In this case, every individual has a probability associated to its fitness value. In this assignment, the individuals with a higher fitness value will be less likely to be eliminated, but there is a chance that they may be removed from the new population. In this selection method, the weaker individuals have a chance of surviving the selection process and becoming members of the new population (SivaraJ & Ravichandran Dr., 2011).
4.1.4.3 Tournament Selection

The tournament selection method chooses two individuals or chromosomes from the initial population. After the selection, their fitness values are compared, and the winner will be the one with the highest fitness value. Tournament selection can include more than two individuals to be selected in order to compete, and the best ones will be able to reproduce. However, a common tournament size is two (binary tournament). When the tournament size is adjusted, the selection pressure can be large or small because the below average individuals have less probability of winning the tournament, but the ones with high fitness are more likely to win (SivaraJ & Ravichandran Dr., 2011).

Figure 15: Tournament selection

4.1.4.4 Steady State Selection

The steady state selection basically eliminates the worst chromosomes or individuals of the population and works as follows: during every generation, only few high fitness individuals are selected
for reproduction, and the bad ones are eliminated. Thus, the new offspring becomes part of that position, and the rest of the population survives for future generations (Mitchell, 1999).

4.1.4.5 Elitism

Elitism is a selection method that basically retains some number of the best individuals during each generation. This method improves the selection process and saves the best chromosomes for reproduction. Consequently, the quality of the solutions during each generation will increase over time (Mitchell, 1999).

4.1.5 Parameter Values

The most challenging aspect, when using Gas, lays in the proper selection of parameter values such as in the population size, crossover, and mutation rates (Boyabatli & Sabuncuoglu). These designated values are very important in the algorithm performance. They are usually used to control the runs of the algorithm and can influence the reproduction and population portion (Sarmandy).

4.1.5.1 Population Size

One of the most important parameter values to select when constructing a genetic algorithm is to determine the initial population size that will be used. The value assigned represents the number of individuals or chromosomes that will cover the population. While working with larger population sizes, the amount of variation is greater and demands more fitness calculations. A good population must include a diverse selection, causing, as a result, a better exploration. Thus, increasing the population size will raise the GA accuracy. Different researchers, such as De Jong, suggest using a population size of 50 to 100 individuals; he reached that conclusion after performing different experiments (De Jong, 1975). Another researcher, Grefenstette, recommended a population of 30 chromosomes (Grenfenstette, 1986). However, Schaffer indicated in their studies that the population size should be between 20 to 30 chromosomes (Schaffer, Caruana, Eshelman, & Das, 1989).
4.1.5.2 Crossover Rate

The crossover rate decides what probability of crossover will happen in the algorithm. As explained before, the crossover operator will reproduce new individuals by combining two chromosomes, or individuals, of the initial population by using some of the crossover techniques stated earlier through the elaboration of the thesis. At the present time, there is not a general rule that could indicate the proper crossover rate that can be used in problems approached with GAs. Only through experimentation is possible to compare the proper combination of the parameter values. For example, De Jong suggested, through his investigations, to use a crossover rate of 0.6 per pair of parents (De Jong, 1975). On the other hand, Grefenstette proposed to use a crossover rate of 0.95 (Grenfenstette, 1986). However, Schaffer concluded, after a year of testing a different range of parameter combinations, that the best crossover rate could be a value within the range of 0.75 to 0.95 (Schaffer, Caruana, Eshelman, & Das, 1989).

4.1.5.3 Mutation Rate

The mutation rate determines the probability that the mutation will happen, and it is applied in order to provide new information to the population and prevents that the new population from beign formed by similar chromosomes or individuals. Since there is not a fixed mutation value that can be used when working with GAs, it is also necessary to take into account the different studies that already exist and then compare the algorithm performance through experimentation. For instance, the proposed mutation rate of De Jong is 0.001 (De Jong, 1975). Contrary to the recommendations of De Jong; Grefenstette recommended using a mutation rate of 0.01, and Schaffer indicated that the best mutation rate occurs when using a value between the range of 0.005 and 0.01 (Grenfenstette, 1986).
4.1.6 Applications

The application of GAs is diverse due to this methodology and can be adapted to several schemes which may be used to solve a wide range of problems. The powerful characteristic of this evolutionary algorithm comes from the fact that it is a robust technique which can deal effectively with different subjects. Thus, its application is mainly in difficult problems where there are no other solving techniques for them.

Genetic Algorithms have been used in different areas such as:

- **Numerical Optimization**: In numerical optimization area, GAs are able to outperform conventional optimization techniques, such as, discontinuous, multimodal, and noisy functions (Beasley, Bull, & Martin, 1993).

- **Combinatorial Optimization**: Also, in the combinatorial optimization part there are different applications such as with the travelling salesman problem (it involves the task of finding the shortest route for visiting a specified group of cities), bin packing (it determines how to locate several objects into a limited space), job shop scheduling (the mission is to allocate efficiently the resources to be able to accomplish a task such as machines, people, rooms, facilities, etc.), and so forth (Beasley, Bull, & Martin, 1993).

- **Image Processing**: In the medical area, when taking medical X-rays or satellite images, there is a necessity of aligning two images of the same area which need to be taken several times. The contribution of GAs in this type of problem is to find a set of equations which can convert one image to adjust into the other; this can be done by comparing randomly a sample of points on the two images. Another unusual application is with the production of images of pictures of criminal suspects in an investigation. In this case, the GA is placed in the role of traditional photo-fit system but has a similar coding scheme. In this application, the GA randomly generates a number of faces, and the witness will
select two pictures that are similar to the suspect’s face. In this case, the witness acts as the fitness function and is able to control the convergence on the way to the correct image (Beasley, Bull, & Martin, 1993).

- **Design:** These types of problems may involve a mix of combinatorial and function optimization problems. Some design applications are: bridge structure, a fire hose nozzle, and neural network structure (Beasley, Bull, & Martin, 1993).

- **Machine Learning:** Genetic Algorithms have been used for many machine learning applications, which include the classification and task prediction. For instance, the prediction of dynamical systems, weather prediction, and also in the field of control, such as in the chemical plants where there may be many control parameters to be adjusted to keep the system running in an optimal way (Beasley, Bull, & Martin, 1993).

The listed applications of GAs previously explained just shows a little portion of approaches in which GA can be adapted. However, there are others that are not discussed, but this information can give a taste about the kind of things a GA is able to do.

### 4.2 Bee Colony Optimization

The Bee Colony Optimization (BCO) metaheuristic pertains to the Swarm Intelligence (SI) group. This group belongs to the social insects that are flexible and can adapt well to the changing environment (bees, ants, wasps, termites). One important characteristic of the SI groups is that they are formed by social groups which share a collective intelligence. This collective intelligence is based according to the group of actions and interactions within their colony. The discussed interactions can be characterized from different chemical or physical signals causing as a result the representation of the social insect colony behavior. Consequently, many nature-inspired algorithms have been motivated due to the characteristic behavior of some of those groups (Teodorovic & Dell'Orco, Bee Colony Optimization - A cooperative Learning Approach to Complex Transportation Problems). BCO and basic
concepts were proposed during the 1999 to 2001 years by Lucic and Teodorovic while doing research in Virginia Tech. BCO is metaheuristic because it represents a general algorithmic framework suitable for various optimization problems. As stated before, BCO pertains to the class of nature- and population-based inspired algorithms. The BCO algorithm simulates the way in which optimization algorithms search for optimal solutions using combinatorial optimization problems. The main purpose when using BCO is to create a multi-agent system that can be useful to solve very complex optimization problems (Teodorovic & Selmic, Bee Colony Optimization: The Applications Survey, 2011).

4.2.1 Bees’ Natural Behavior

The bee colony in reality is formed by approximately 60,000 to 80,000 bees in the hive. A colony includes a single queen, thousands of males (called drones), other thousands of females (called workers), and many young larvae (called broods). Their primary food source is the nectar, which is used as a source of energy, and they also use pollen in order to provide proteins to the little larvae. The communication between the bees of the colony is very precise and is based mainly in dances. These types of dances are performed by the scout bee. The scout bee is the one who explores the area in order to look for food. Once the explorer bee (scout) finds food and comes back to the hive, the scout bee informs the other bees about the quality, distance, location, and quantity of the food found. Consequently, the other bees perceive by their visual, tactile, and olfactory perception the information transmitted by the scout bee. There are two different types of dances that are performed by the bees. One is the round dance, which the bee can realize in order to indicate that the food is relatively close. The second dance is called the waggle dance; when the bee dances this type of dance, the bee forms the shape of the number eight. This movement form indicates the distance and direction of the food source. To be able to indicate to the other bees the distance of the food found, the scout bee will dance in a certain speed. Consequently, if the dance is faster, it means that the distance to reach the food is smaller and longer otherwise. Then, to indicate in what direction the rest of the bees should go to find the food,
the bee who is dancing shows a certain inclination while dancing. This inclination forms an angle between the food source and the sun relative to the hive. The quality of the food is represented by the scent of the bee when it is rubbed. To characterize the quantity of the food this basically depends on the wriggling of the bee. If the wriggling is abundant, it means that there is more food and less otherwise.

![Diagram of honeybee dance](https://example.com/diagram.png)

**Figure 16:** Bee waggle and round dance  
Source: Diagram of the honeybee dance. (Credit: P. Kirk Visscher.)

During the early spring, some queen cells are produced to be able to generate a new queen. Before the birth of the second queen, the first queen leaves the colony. The leaving queen, however, leaves the hive in company of other bees in order to form a new colony. When looking for a possible nest site, the scout bees will seek different nest areas. Finding a new area, the scout bees will perform the appropriate dances in order to indicate the various locations and qualities of the nests.

When searching for food, some scout bees explore the region. Once they return to the hive, the scout bees perform the dances to transmit the location of the discovery to the others. Some bees will be recruited to become foragers. This step is called the exploration phase and then follows with the exploitation step. When the bees gather the food, they calculate the food quantity in order to make a new decision. That decision includes whether to continue collecting the food found or to leave it and return to the hive as a simple bee.

The bee’s reproduction is a phenomenon guaranteed by the queen. The queen will mate with several males while flying, and after three days, the queen will lay eggs. The unfertilized egg will raise a
drone (male), and the fertilized egg will raise either a worker (female) or a queen. To be able to give raise to a queen, it will depend on the food quality provided to the larvae (Bitman, Mohamed, & Talbi, 2010).

4.2.2 Bees’ Artificial Behavior

The BCO is grouped by a colony of artificial bees which live in an environment characterized by a discrete time. This colony has a population of $N$ bees that collaborate in the search of an optimal solution for a determined problem. In BCO, each artificial bee generates a solution in the problem. The algorithm includes two alternating phases such as forward and backward pass. Both phases represent a single step in the elaboration of the BCO. During the forward pass, each artificial bee visits certain $NC$ (number of constructive moves) solution components in order to create a partial solution and after that the bee returns to the hive.

![Figure 17: Example of partial solutions after the first forward pass, NC=3, N=3](source: Empirical study of the bee colony optimization (BCO) algorithm)

The hive in the BCO algorithm does not have a specific location and does not affect the algorithm performance. It just represents a synchronization point in which the bees can exchange information about the solutions of the exploration. The numbers of components $NC$ to be visited by the bee (predefined number of moves in order to construct a solution) are indicated by the user since the beginning of the exploration process (forward pass). For instance, Figure 17 represents the forward pass...
when there is \( N \) number of bees. In this particular illustration, \( N \) is equal to three bees, and the number of components (\( NC \)) is equal to three as well. In this example, at each forward pass, the bees are supposed to have visited the three components of the search space.

Once the bees have completed the initial partial solutions of the problem (partial solutions are obtained during the forward pass), the bees return to the hive in order to communicate their findings to the rest of the bees. The time when the communication process about those results begins is considered to be the backward pass (Figure 18). As soon as all the partial solutions have been communicated and evaluated by the rest of the bees in the hive, each bee will decide either to keep its solution or not. This decision will be based according to a certain probability value. However, the bees that have found the better solutions have more probability of keeping their results and will continue exchanging that information to the rest of the bees. This type of artificial bee loyal to its solutions is called recruiter bees because they are recruiting more bees at the same time. Nevertheless, if a bee decides to abandon a solution, it becomes uncommitted but will select another advertised solution to follow. Also, this decision is taken primarily with a certain probability. Consequently, the better advertised solutions will have a greater possibility to be selected and explored. Therefore, through each backward pass all the bees are divided into two different groups such as recruiters (\( R \)) and the rest of the bees (\( R-N \)) will be called the uncommitted bees (Figure 19) (Teodorovic & Selmic, Bee Colony Optimization: The Applications Survey, 2011). While moving from one backward pass to another, the \( R \) and \( R-N \) values will be changing. For instance, when the bees have compared all the partial generated solutions, \( N=3 \) (bee three) from the previous example has decided to abandon its previous solution and join \( N=1 \) (bee one). Both bees will fly together along the path founded by bee one. This means that the partial solution generated by bee one will be copied to bee three. However, when they have reached the end of the path, they will make an individual decision about their next step. This means that both bees will add a different component to the same partial solution. However, bee two will keep its same solution without
having been chosen by any hive-mates and will elaborate a new constructive step independently (Figure 20). The two phases described previously, forward and backward pass, alternate to get all the required complete solutions, and an iteration of the BCO will be completed. After the algorithm completes all the iterations, it will stop, and the best solution found will be reported at the end (Nikolic & Teodorovic, 2013).

Figure 18: The first backward pass, NC=3, N=3
Source: Empirical study of the bee colony optimization (BCO) algorithm

Figure 19: Dividing bees into two groups, N=3
Source: Empirical study of the bee colony optimization (BCO) algorithm
4.2.3 BCO Algorithm Pseudo-code

The algorithm parameters that need to be set to initialize the BCO are the \( N \) number of bees in the hive and the \( NC \), the number of constructive moves during one forward pass. It is assumed that at the beginning of the exploration search, all the bees are waiting in the hive. Next there is the pseudo-code of the BCO algorithm:

1. **Initialization:** In this step every bee will be assigned to an empty solution.
2. For each bee that does the forward pass:
   a. Set \( k=1 \); \( k \) is a counter that indicates the constructive moves during the forward pass.
   b. An evaluation of the possible constructive moves will be realized.
   c. After getting the evaluation, every bee will decide what to use by using the roulette wheel selection method.
   d. \( k=k+1 \); If the counter \( k \) is less or equal to the number of constructive moves \((NC)\) then continue performing step b.
3. During this step, all the bees have returned to the hive, and the backward pass begins.
4. The partial objective function is evaluated for each bee.
5. Every bee will decide randomly whether to continue its own exploration or become a recruiter or a follower. In this step, the bees that got the highest fitness value will have more possibility of continuing its own exploration.

6. The follower bees will select a new solution to follow from the recruiters by using the roulette wheel selection method.

7. If the solutions are not completed until this phase, then go to the second step.

8. Evaluate all the solutions obtained and then go to step one.

9. If the algorithm has not reached the stopping criteria, then go to step 2.

10. At the end the program indicates the best solution found.

4.2.4 BCO Applications

Next, there is a table showing the algorithms that have been inspired by the bee’s behavior, including the year and their authors’ names.

<table>
<thead>
<tr>
<th>Year</th>
<th>Authors</th>
<th>Algorithm</th>
<th>Source: Bee colony optimization (BCO)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1996</td>
<td>Yonezawa and Kikuchi</td>
<td>Ecological algorithm</td>
<td>Description of the collective intelligence based on bees’ behavior</td>
</tr>
<tr>
<td>1997</td>
<td>Sato and Hagiwara</td>
<td>Bee System (BS)</td>
<td>Genetic Algorithm Improvement</td>
</tr>
<tr>
<td>2001</td>
<td>Lučić and Teodorović</td>
<td>BCO</td>
<td>Traveling salesman problem</td>
</tr>
<tr>
<td>2001</td>
<td>Abbas</td>
<td>MBO</td>
<td>Propositional satisfiability problems</td>
</tr>
<tr>
<td>2002</td>
<td>Lučić and Teodorović</td>
<td>BCO</td>
<td>Traveling salesman problem</td>
</tr>
<tr>
<td>2003</td>
<td>Lučić and Teodorović</td>
<td>BCO</td>
<td>Vehicle routing problem in the case of uncertain demand</td>
</tr>
<tr>
<td>2004</td>
<td>Wedde, Farooq, and Zhang</td>
<td>BeeHive</td>
<td>Traveling salesman problem</td>
</tr>
<tr>
<td>2004</td>
<td>Teodorović, and Dell’ Orco</td>
<td>BCO</td>
<td>Routing protocols</td>
</tr>
<tr>
<td>2005</td>
<td>Karaboga</td>
<td>ABC</td>
<td>Ride-matching problem</td>
</tr>
<tr>
<td>2005</td>
<td>Drias, Sadeg, and Yahi</td>
<td>BSO</td>
<td>Numerical optimization</td>
</tr>
<tr>
<td>2005</td>
<td>Yang</td>
<td>Virtual Bee Algorithm (VBA)</td>
<td>Maximum Weighted Satisfiability Problem</td>
</tr>
<tr>
<td>2005</td>
<td>Benatchba, Admane, and Ksoudil</td>
<td>MBO</td>
<td>Function optimizations with the application in engineering problems</td>
</tr>
<tr>
<td>2006</td>
<td>Teodorović, Lučić, Markovlić, and Dell’ Orco</td>
<td>BCO</td>
<td>Max-Sat problem</td>
</tr>
<tr>
<td>2006</td>
<td>Cheng, Low, Sivalumar, and Gay</td>
<td>Honey Bee Colony Algorithms</td>
<td>Traveling salesman problem and a routing problems in networks</td>
</tr>
<tr>
<td>2006</td>
<td>Pham, Soroja, Ghanbarzadeh, and Koe</td>
<td>Bees Algorithm</td>
<td>Job shop scheduling problem</td>
</tr>
<tr>
<td>2006</td>
<td>Rustem and Karaboga</td>
<td>ABC</td>
<td>Optimization of neural networks for wood defect detection</td>
</tr>
<tr>
<td>2006</td>
<td>Navrat</td>
<td>BeeHive Model</td>
<td>Numeric function optimization</td>
</tr>
<tr>
<td>2006</td>
<td>Wedde, Timm, and Farooq</td>
<td>BeeHiveAIS</td>
<td>Web search</td>
</tr>
<tr>
<td>2007</td>
<td>Yang, Chen, and Tu</td>
<td>MBO</td>
<td>Routing protocols</td>
</tr>
<tr>
<td>2007</td>
<td>Ksoudil, Benatchba, Tarabetand, and El Batoul</td>
<td>MBO</td>
<td>Improvement of the MBO algorithm</td>
</tr>
<tr>
<td>2007</td>
<td></td>
<td></td>
<td>Partitioning and scheduling problems</td>
</tr>
</tbody>
</table>
The general Bee Colony Optimization algorithm has various applications in several problem areas (Table 1). It has been used to solve certain benchmark problems such as the Travelling Salesman Problem, routing problems, and NP-hard problems. Following is a brief description about the different areas in which BCO has been applied.

- **Solving complex transportation problems**: The BCO Algorithm has been used to solve the transportation problem using the ride-matching problem. In this investigation, the BCO concept is about the social insects’ behavior and their flexibilities in these types of colonies. It also discusses the bee’s collaboration in order to find solutions to different combinatorial optimization problems.
• **Fault based test suite prioritization:** This study presents a BCO algorithm in order to prioritize a regression test suite on maximum fault coverage. The authors of this work discuss the natural bee system in detail. Here the scout and forager’s food scour scenarios were mapped to prioritize the test suite. This well organized algorithm was designed and developed in two parts (for foragers and scouts).

• **Solving sudoku:** Another application of the BCO was developed to solve sudoku puzzles which are classified as NP-hard problems. A sudoku is a logical 2-D array in row, column, and diagonal without repetition. Basically, what this algorithm does is to mimic the method through which bees forage for food.

• **Problem solving mechanism:** In this research area the bees behavior within and outside the hive have been separated in order to solve problems that are involved in different types of domains. Some of those domains are web searching, function optimization, and hierarchical optimization problems.

• **Accident diagnosis:** In this investigation, the algorithm performed a random neighborhood search. Then, the algorithm found the best solution and identified an accident in a nuclear power plant. After, a comparison and analysis of the proposed algorithm in this research was found to be more efficient after comparing it with other population-based search algorithms. The software used to solve the BCO algorithm was done with Matlab.

• **Maximum satisfiability problem:** Another application of the BCO is to solve a maximum weighted satisfiability problem. This type of problem is considered a NP-hard problem. In this particular case the proposed algorithm was implemented and proved on certain well known benchmark problems.
• **Job shop scheduling:** In the manufacturing area, BCO has been applied for the job shop scheduling problem. Job shop scheduling is vital for the manufacturing industry because it helps to improve the machine utilization and reduces the cycle time.

• **Travelling salesman problem:** The Travelling Salesman Problem is a very well-known problem which has been solved with different algorithms. However, this problem has also been solved with BCO. There are other detail analyses that have been done to compare the efficiency of BCO with other approaches (Teodorovic & Dell'Oro).

### 4.3 Summary

This chapter presents the two metaheuristics used in this investigation which are Genetic Algorithms and Bee Colony Optimization. There is also a full description about the GA’s operators which are crossover, mutation, and its selection methods. Furthermore, metaheuristics explain the various encoding types that can be used in Gas, according to the problem type. It also includes detailed information about the most common crossover, mutation, and selection techniques. The chapter presents a section talking about certain studies realized in order to determine the best parameter values to use in Gas. However, there is not a general rule that indicates a fixed value to use. Moreover, a section about the GA’s applications is shown to demonstrate some of the important applications that this algorithm may have. Another part of the chapter introduces the BCO, which has been developed recently. Next, an explanation about the bees’ natural behavior helps to understand clearly the amazing behavior of this colony system. After, the general BCO algorithm is described along with its pseudo-code. In BCO it is necessary to understand how the artificial bees work in that environment to be able to understand the BCO functionality. Also, some applications of BCO are mentioned just to have an idea about its importance.
CHAPTER 5: PROPOSED METHODOLOGY FOR GA AND BCO

GA and BCO are important metaheuristic techniques that have been applied to a great diversity of problems. However, there are other problems in which neither of them has been tested. This is the situation of the problem under discussion in this thesis. The following chapter explains how the logistic system design problem is implemented to the GA and BCO methodology.

5.1 GA Approach

The investigation in this thesis work is based in the logistic system design problem described in Chapter 3. The next figure shows a schematic diagram which indicates the problem flow.

![Figure 21: GA schematic diagram for logistic system design problem](image-url)
Following is the explanation about every section involved in the schematic diagram:

1. **Chromosome elaboration:** The encoding of the biomass logistic design problem is a big challenge because the chromosome should be able to provide all the information required at the end. As explained in Chapter 4, there are different encoding techniques that can be applied such as binary, permutation, and tree encoding. However, in this particular problem, it is necessary to use three encoding types, for example, the binary, permutation, and real value encoding due to it being more suitable to the problem data. The next figure represents the chromosome generated for the problem explained.

<table>
<thead>
<tr>
<th>Months</th>
<th>3</th>
<th>9</th>
<th>7</th>
<th>5</th>
<th>6</th>
<th>10</th>
<th>12</th>
<th>11</th>
<th>1</th>
<th>8</th>
<th>4</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biomass Type</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Field Origin</td>
<td>13</td>
<td>7</td>
<td>8</td>
<td>14</td>
<td>12</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>Tons of Biomass Processed</td>
<td>112981</td>
<td>79055</td>
<td>61901</td>
<td>96557</td>
<td>104546</td>
<td>102022</td>
<td>97552</td>
<td>70694</td>
<td>65531</td>
<td>31906</td>
<td>102637</td>
<td>91822</td>
</tr>
<tr>
<td>Biorefineries Open</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Tons of Residue produced</td>
<td>2259.62</td>
<td>1581.1</td>
<td>1238.02</td>
<td>1931.1</td>
<td>2000.9</td>
<td>2040.44</td>
<td>1951</td>
<td>1412.1</td>
<td>13102</td>
<td>638.12</td>
<td>2052.7</td>
<td>1836.4</td>
</tr>
<tr>
<td>Residue Recirculated to Field</td>
<td>0</td>
<td>1</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>10</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>In-Field Warehouse</td>
<td>0</td>
<td>1</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>10</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>Tons of Residue Storage</td>
<td>2259.62</td>
<td>1581.1</td>
<td>1238.02</td>
<td>1931.1</td>
<td>2000.9</td>
<td>2040.44</td>
<td>1951</td>
<td>1412.1</td>
<td>13102</td>
<td>638.12</td>
<td>2052.7</td>
<td>1836.4</td>
</tr>
<tr>
<td>Type of Biomass Stored</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Biomass Type</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>From Field Origin</td>
<td>13</td>
<td>7</td>
<td>8</td>
<td>14</td>
<td>12</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>Intermediate Warehouse Open</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Possible Intermediate Warehouse Open</td>
<td>0</td>
<td>16</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Transportation Type</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>To Biorefinery</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Transportation Type</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Tons of BiomassTransported</td>
<td>112981</td>
<td>79055</td>
<td>61901</td>
<td>96557</td>
<td>104546</td>
<td>102022</td>
<td>97552</td>
<td>70694</td>
<td>65531</td>
<td>31906</td>
<td>102637</td>
<td>91822</td>
</tr>
<tr>
<td>Biomass Type</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>From Biorefinery</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>To Switchgrass Fields</td>
<td>10</td>
<td>1</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>10</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>Transportation Type</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Tons of Residue Transported</td>
<td>2259.62</td>
<td>1581.1</td>
<td>1238.02</td>
<td>1931.1</td>
<td>2000.9</td>
<td>2040.44</td>
<td>1951</td>
<td>1412.1</td>
<td>13102</td>
<td>638.12</td>
<td>2052.7</td>
<td>1836.4</td>
</tr>
</tbody>
</table>

The chromosome obtained for the logistic system design problem, including three different types of biomass feedstock, is composed of five different sections. The first section represents important information about the biomass that will be processed in both biorefineries during the year. The biomass type is part of the first section of the chromosome and is represented with the numbers 1, 2, and 3, where 1 stands for switchgrass, 2 is stalk, and 3 means straw. In the same section, the months are generated randomly from one to twelve, but it is important to remember the fact that during the months March, April, May, and June, the switchgrass is not harvested. Consequently, whenever the chromosome generates the months three, four, five, or six, it means that only the biomass stalk or straw can be
purchased. Since the problem deals only with fourteen fields where the biomass feedstock is obtained, the field of origin can be a number from one to fourteen. The dry tons of biomass processed in the biorefineries should not exceed the 2,100,000/year. During the year each biorefinery could be open or closed in the whole year or it could be closed only for certain periods of time, it was decided to operate both biorefineries during the entire year to be able to maximize the biofuel production.

The second part of the chromosome indicates the amount of residue produced during each month in the biorefineries. That residue will be transported to the fields because it serves as a fertilizer to the land and to keep the nutrients of the soil. Nevertheless, the residue will be taken only to the switchgrass fields because they are part of the system, which is why the chromosome only includes in that particular section values from one to ten.

The third section of the chromosome represents the amount of residue that will be stored in the in-field warehouses and also indicates the type of biomass that will be stored.

The section number five indicates how the biomass since is transported once it is harvested. First of all, what the chromosome does in this area is indicate what type of and where the field of origin of the biomass is harvested to know the transportation type that will be needed. It also indicates whether an intermediate warehouse will be open or not. With the binary encoding, it is noticeable if the intermediate warehouses are open or not (1=open; 0=close). If the intermediate warehouse is open, then it will need a transportation mode such as train or truck that will depend in the location of the field of origin. However, if the intermediate warehouse is not open during that month, then the biomass will be transported directly from the field of origin to the biorefineries.

Finally, the fifth section of the chromosome includes how and the amount of residue to be transported from the biorefineries to the switchgrass fields.

2. **Population size:** The size of the population established in the problem is 50 individuals or chromosomes.
3. **Generations:** The number of generations, or iterations, is established by the user. In this particular problem, the numbers of iterations realized in the algorithm were 100.

4. **Evaluate fitness function:** After a random initial population is generated, every chromosome will generate the revenue, all the costs involved in the problem, and its profit.

5. **Rank selection:** The selection method used in this problem is the rank selection method. What this selection method does is to rank the profit obtained from every chromosome and sort them from the best to the worst value. The fitness values are obtained with the objective function of the problem, which in this case is to maximize profit. After, the values are sorted and only 70% of those results are selected. The reason why this selection method is used instead of using elitism is because with rank selection it is possible to prevent very fit individuals or chromosomes from getting more control too early in the process not allowing the diversity in the population and having problems to find an acceptable solution.

6. **Reproduction process:** The crossover technique used in the chromosome realized is single crossover. The only sections where the single crossover took place are the dry tons of biomass processed, whether the intermediate warehouses are open or not, and the production fields where the residue is recirculated. For the rest of the chromosomes, there are sections where it is impossible to apply the crossover part because most of the information in the chromosome is closely related and cannot be easily interchanged. Thus, to maintain the integrity of the chromosome only the mentioned values were interchanged. The percentage of the population that received crossover is 70%.

7. **Mutation:** The mutation rate applied in the problem was 1%. This was done in order to originate new members in the population and to prevent the formation of similar chromosomes or individuals.

8. **Evaluate new population:** After the reproduction process and mutation takes place, new chromosomes will become part of the new population which needs to be evaluated.
9. There are more generations, or iterations, to compute:
   a. If yes; then go to step 4.
   b. If no; then store final solution.

5.2 BCO Approach

This section includes the explanation about how the logistic system design problem could be adapted to the BCO metaheuristic. The following figure shows the schematic diagram about BCO applied to the problem:

![BCO Schematic Diagram](image-url)

Figure 22: BCO schematic diagram for logistic system design problem
Following is the explanation about every part in the BCO schematic diagram:

1. **Parameters:** At first, the number of bees must be indicated by the user. In this case, the number of bees established is 12. Also, the number of constructive moves needs to be indicated by the user at the beginning of the problem. Those movements are the ones that a bee needs to do in order to create a path. The twelve movements in the problem represent the number of months in the year, which is the time frame used in the problem under discussion.

2. **Random preferred path:** Initially, the bees are placed randomly in a preferred path, and they are waiting to start the exploration process.
   a. The length of the preferred path is twelve squares, which represents the months of the year. Every square will be completed by a number, and that value will represent one of the four possibilities explained below:

   **Option one:** Option number one will contain important information about which one will be the field of origin where the biomass will be obtained. The fields can be a number from 1 to 14 because the network flow includes 1 to 10 fields of switchgrass, 11 and 12 of the wheat straw, and 13 and 14 of the corn stalk. Also, by knowing the number of the field, the type of biomass feedstock will be identified. Furthermore, in option one, there are not intermediate warehouses open. The reason is because the wheat straw and corn stalk will go directly from the field of origin to the biorefineries and, due to the switchgrass, may not be stored in the intermediate warehouse during that particular month. The biomass here will be transported only from the field of origin to the two biorefineries open. Then, the residue generated in the biorefineries will be transported from the biorefineries to one of the ten switchgrass fields. The transportation modes used can be either by truck (trucks=1) or by train (train=2).

   **Option two:** The second option also indicates the field of origin (may be from 1-14). However, in this possibility, one of the three intermediate warehouses will be open. In this case
the intermediate warehouse number 15 will be open. Since that warehouse is now open, the biomass feedstock will be transported from the field of origin to the intermediate warehouse number 15. Then, it will go from the intermediate warehouse to the biorefineries. After that, the residue will be moved from the biorefineries to one of the switchgrass fields (may be from 1-10). The biomass will be transported by using the trucks or the train.

Option three: The third option also indicates the field of origin (may be from 1-14). However, in this possibility one of the three intermediate warehouses will be open. In this case the intermediate warehouse number 16 will be open. The biomass feedstock will be transported from the field of origin to the intermediate warehouse number 16. Then, it will go from the intermediate warehouse to the biorefineries. After that, the residue will be relocated from the biorefineries to one of the switchgrass fields (may be from 1-10). The biomass will be transported by using the trucks or the train.

Option four: The fourth option also represents the field of origin (may be from 1-14). However, in this possibility one of the three intermediate warehouses will be open. In this case the intermediate warehouse number 17 will be open. The biomass feedstock will be transported from the field of origin to the intermediate warehouse number 17. Then, it will go from the intermediate warehouse to the biorefineries. After that, the residue will be relocated from the biorefineries to one of the switchgrass fields (may be from 1-10). At that moment, the biomass will be transported by using the trucks or the train.
The preferred paths developed in the algorithm are constructed as follows:

The use of the four options is established in this part of the algorithm, but as in the TSP problem, the cities cannot be repeated; it was decided to represent those options with different values. For instance, if during the second month (February), option one is part of the preferred path, that option value will be represented with the number five (Figure 23). At the end, the preferred path will be constructed by 12 numbers (months of the year) which will not be repeated, and every number will represent one of the four possibilities previously described.

3. **Forward pass:**

   a. **Path elaboration:** Every bee will generate a new path, then it must be evaluated by calculating the seven costs involved in the problem, the revenue, and the profit per month. After getting the profit, the state transition probability formula (Equation 16) is used to help the bee decide its next move from option $i$ to option $j$ at time $t$ (months). The bee will finish the path after it has moved itself twelve times. The probability formula is a function of the profit and the arc fitness presented on the connecting edges. The arc fitness formula is shown in Equation 17:
\[
P_{ij}(t) = \frac{[\rho_{ij}(t)]^\alpha [\text{profit}_{ij}(t)]^\beta}{\sum_{j \in A_i(t)} [\rho_{ij}(t)]^\alpha [\text{profit}_{ij}(t)]^\beta}
\]

(16)

Where:

\[\rho_{ij}(t)\] = Arc fitness value from option \(i\) to option \(j\) at time \(t\).

\(\alpha\) = Parameter value that controls the influence of arc fitness.

\(\text{profit}_{ij}(t)\) = Profit value of the next possible movement from option \(i\) to \(j\) at time \(t\).

\(\beta\) = Parameter that controls the level of profitability.

The arc fitness equation basically is to assign a fitness value to all the arcs that connect to the hive. The first option visited in the preferred path, in this example, is considered to be the hive. The value \(\lambda\) will be assigned to the option connected to the hive, and the other values will have a value determined by \((1- \lambda)|A_i(t)| - |A_i(t) \cap F_i|\). The information \(|A_i(t)|\) represents the allowed next option that the bee has to visit, and the expression \(|A_i(t) \cap F_i|\) represents the position value that connects with the hive. For example, if \(\lambda\) was already assigned the rest of the values, it will be divided by 11, 10, 9… 1 because the total length of the preferred path, in this problem, is 12. The symbol \((\lambda)\) will be a value between 0 and 1 (Wong, Yoke Hean Low, & Soon Chong, 2008).

\[
\rho_{ij}(t) = \begin{cases} 
\lambda \\
\frac{1-|A_i(t) \cap F_i(t)|}{|A_i(t)| - |A_i(t) \cap F_i|} \\
\sum_{j \in F_i(t)} \text{profit}_{ij}(t) 
\end{cases} \forall \in A_i(t), 0 \leq \lambda \leq 1
\]

(17)

If the bees have completed the twelve constructive moves, then the backward pass begins, but if it is not completed, it must continue after obtaining a path conformed by that amount.

The parameter values used in the formulas were the following:
4. **Backward pass:**

   a. **Identify best profit:** After having all the paths formed by the explorer bees, including the twelve values, then partial solutions about the costs (inventory, purchasing, storage, processing, harvesting units, transportation by truck and train costs) analyzed in the problem are calculated, and the revenue is obtained as well. By having the total costs and the revenue, the fitness function, or objective function, is used in order to calculate the profit (Equation 18).

   \[
   \max R - \sum_{n=1}^{7} C_n
   \]  

   Where:

   - \( R \) = The total revenue.
   - \( C_n \) = The total costs.

   b. **Bees’ decision:** Once the paths are completed, the bees with the maximum profit will be able to dance. By dancing they will be able to communicate their findings to the rest of the bees that are in the hive. Then, the bees of the hive will need to evaluate the partial solution offered by the explorer bees and decide, according to the duration of the dance, what path to follow or to look for their own path.

   The dance duration is calculated by Equation 19 where \( K \) is proportionality constant. Also, the duration formula includes the \( Pf_l \) (Equation 20), which represents the best profits of the bees that will dance. And \( Pf_{colony} \) (Equation 21) is the summation of all the best profits that were multiplied individually by \( 1/n \) where \( n \) are the number of bees that are dancing.
Then, once the bee has observed the dance, it has to determine if it will follow the dance shown by the previous dancer with a probability of $P_{\text{follow}}$. The selection method utilized was the roulette wheel method. This selection method is performed every iteration and allows the values with highest probability to have more chance of being chosen. The profitability score of the bee and the colony is adjusted according to the information displayed in Table 4 (Wong & Soon Chong, 2009)

<table>
<thead>
<tr>
<th>Profitability Scores</th>
<th>Pf_follow</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{f_i} &lt; .95 P_{f_{\text{colony}}}$</td>
<td>0.6</td>
</tr>
<tr>
<td>$.95 P_{f_{\text{colony}}} &lt; P_{f_i} &lt; .975 P_{f_{\text{colony}}}$</td>
<td>0.2</td>
</tr>
<tr>
<td>$.975 P_{f_{\text{colony}}} &lt; P_{f_i} &lt; .99 P_{f_{\text{colony}}}$</td>
<td>0.02</td>
</tr>
</tbody>
</table>

5. **Stopping criteria reach:** The stopping criteria are the number of iterations which were established by the user. In this problem the number of iterations is 100.

   a. If it is not; then go to step 1 and start the process again.

   b. If it is; then store best solution found.
5.3 Summary

This chapter presents how the logistic system design problem is applied with the help of GA and BCO. It also includes schematic diagrams for both approaches in order to show the process flow. The GA approach part is described by how many sections the chromosome includes such as biomass processed, residue recirculated, residue storage, feedstock transported, and residue transported. It also indicates the encoding techniques applied in the chromosome or individual made. Furthermore, it explains the single crossover and rank selection method used. Also, the parameter values for both algorithms are indicated. In BCO, the preferred path elaboration is discussed and also explains the four options presented in the problem. Moreover, it introduces the use of the arc fitness and probability formula to help the bees do the next constructive movement, which is conformed by twelve values. Then, after identifying the best profits, the bees with the highest profit will dance. After, the bees will decide whether to continue its own path, becoming a recruiter or a follower. This decision is based on the roulette wheel selection method.
CHAPTER 6: NUMERICAL RESULTS AND DISCUSSION

This chapter indicates the results obtained after solving the logistic problem with GA and BCO. It also shows a comparison between both techniques.

6.1 GA Numerical Results

After applying the parameter values indicated in Chapter 5, the GA results are displayed in the following table.

<table>
<thead>
<tr>
<th>Month</th>
<th>Processing Cost</th>
<th>Purchasing Cost</th>
<th>Inventory Cost</th>
<th>Transportation Truck</th>
<th>Transportation Train</th>
<th>Operating Cost</th>
<th>Harvest Unit</th>
<th>Revenue</th>
<th>Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11,129,400.00</td>
<td>11,129,400.00</td>
<td>244,451.76</td>
<td>890.35</td>
<td>89.04</td>
<td>1,726,666.67</td>
<td>1,450,000.00</td>
<td>$36,059,256.00</td>
<td>$10,378,358.19</td>
</tr>
<tr>
<td>2</td>
<td>11,129,400.00</td>
<td>11,129,400.00</td>
<td>244,451.76</td>
<td>890.35</td>
<td>89.04</td>
<td>1,726,666.67</td>
<td>1,450,000.00</td>
<td>$36,059,256.00</td>
<td>$10,290,213.34</td>
</tr>
<tr>
<td>3</td>
<td>11,129,400.00</td>
<td>11,129,400.00</td>
<td>244,599.16</td>
<td>92,903.03</td>
<td>91,686.99</td>
<td>1,726,666.67</td>
<td>1,450,000.00</td>
<td>$37,253,196.00</td>
<td>$14,192,597.14</td>
</tr>
<tr>
<td>4</td>
<td>11,347,400.00</td>
<td>8,048,530.00</td>
<td>244,538.96</td>
<td>91,686.99</td>
<td>91,983.2</td>
<td>1,726,666.67</td>
<td>1,450,000.00</td>
<td>$36,765,576.00</td>
<td>$13,962,103.38</td>
</tr>
<tr>
<td>5</td>
<td>11,497,900.00</td>
<td>8,048,530.00</td>
<td>244,538.96</td>
<td>91,983.2</td>
<td>91,983.2</td>
<td>1,726,666.67</td>
<td>1,450,000.00</td>
<td>$36,765,576.00</td>
<td>$14,275,382.02</td>
</tr>
<tr>
<td>6</td>
<td>11,335,900.00</td>
<td>7,935,130.00</td>
<td>244,534.36</td>
<td>90,687.20</td>
<td>90,687.20</td>
<td>1,726,666.67</td>
<td>1,450,000.00</td>
<td>$36,728,316.00</td>
<td>$14,026,925.57</td>
</tr>
<tr>
<td>7</td>
<td>11,335,900.00</td>
<td>11,335,900.00</td>
<td>11,335,900.00</td>
<td>11,335,900.00</td>
<td>11,335,900.00</td>
<td>1,726,666.67</td>
<td>1,450,000.00</td>
<td>$36,728,316.00</td>
<td>$10,544,537.09</td>
</tr>
</tbody>
</table>

The examination of the problem about having multiple types of feedstock (switchgrass, wheat straw, and corn stalk) resulted in an optimal profit of $135,404,783.58 million. The best profit indicated here is obtained during the iteration number 100 and a unit profit of $0.57 per gallon. When producing an amount of 237,310,740 million gallons of biofuel, the presented profit can be obtained. The revenue generated in this solution is $427,159,332.00. Also, the table includes information about all the costs considered in the problem, such as the processing, purchasing, inventory, operating, harvest units, and transportation by truck and train costs. However, of all the costs analyzed in the problem, the ones who represent a bigger cost are the processing cost with $131,839,300, which represents a 45.188% of all the
costs and the purchasing cost with $118,135,570, which is a 40.491% of the costs (Figure 22). The smallest annual cost in the problem was the transportation cost by train with $27,797.11.

The best individual or chromosome were these results were obtained does not include any intermediate warehouse open because that will mean more additional costs in the problem. Thus, while the algorithm is advancing iteration by iteration, the intermediate warehouses are disappearing over time. The harvesting units needed in order to harvest the 861,602 dry tons/year of switchgrass feedstock were 30. The total amount of residue generated in the biorefineries was 26,367.86 dry tons/year. This amount constitutes the .005% of residue that should be recirculated to the switchgrass fields per year. Next figure represents the percentage of residue generated by month out of the total residue produced by year.

Figure 24: GA annual costs

Operating Cost 7.102%
Transportation Train 0.010%
Transportation Truck 0.240%
Inventory Cost 1.005%
Processing Cost 45.188%
Harvest Unit 5.964%
Purchasing Cost 40.491%
6.2 GA Sensitivity Analysis

A sensitivity analysis is shown below in Table 6 in order to determine what parameter values provide a better result to the logistic system design problem. In GAs, the important parameters are the crossover, mutation, and population size. However, as explained before, at the present time there is not a general rule that specifies any fixed parameter values that can be used to solve any type of problems with GAs. Thus, it is necessary to realize a sensitivity analysis to test different parameter values. When the GA was initially run, a population of 50 individuals was established. The mutation rate was set to 1%. The crossover rate was 70%, and the number of iterations was 100. In the sensitivity analysis the algorithm was tested, using a population size of 20 to 115 individuals or chromosomes. The number of iterations or generations was from 50 to 1000. The crossover rate was from 60% to 79%, and the mutation rate was tested from values from 1% to 2%. As shown in Table 6, the largest profit was obtained when using a population size of 115, a crossover rate of 79%, a mutation rate of 2%, and a generation of 1000. This resulted in an overall profit of $148,492,492 with revenue of $466,528,896. When the algorithm was checked with the first parameter values, the overall profit obtained was of $135,404,783.58 million and yearly revenue of $427,159,332.00. The previous amount is below $13,087,708 million of the new quantity resulted after applying the sensitivity analysis. However, the
computation time to get this solution was time consuming resulting in 604.45 seconds, which is approximately 10 minutes. The algorithm code was programmed with Matlab and performed in a computer with a third generation core i7 processor.

6.3 BCO Numerical Results

The results obtained after running the BCO with the parameters established previously are explained in this section. The analysis of the problem, including the three different types of biomass feedstock’s, provided an optimal profit of $81,432,993.96. This result was produced during the iteration number 85, and $1.04 is the unit profit per gallon. By producing a total amount of 78,314,580 gallons of biofuel, the profit presented can be obtained. The revenue in this result is $281,932,488. Table 7 shows the total costs involved in the problem (processing, purchasing, inventory, operating, harvest units, and transportation by truck and train costs), and it is noticeable that the processing costs is the greatest cost of the problem with $87,016,200, which represents a 43.400% of all the costs, and in second place is the purchasing cost with $78,027,150, which includes the 38.916% of the costs (Figure 26). The smallest cost of the problem was the transportation costs by train with $10,026.82.

Table 6: GA sensitivity analysis

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Population Size</th>
<th>Crossover Rate</th>
<th>Mutation Rate</th>
<th>Computational Time (S)</th>
<th>Best Profit</th>
<th>Iteration</th>
<th>Revenue</th>
<th>Biofuel (gal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>20</td>
<td>0.6</td>
<td>0.01</td>
<td>15.37</td>
<td>$14,200,000</td>
<td>50</td>
<td>$369,271,224</td>
<td>205,150,680</td>
</tr>
<tr>
<td>100</td>
<td>25</td>
<td>0.61</td>
<td>0.002</td>
<td>34.13</td>
<td>$144,632,321</td>
<td>100</td>
<td>$455,496,372</td>
<td>253,053,540</td>
</tr>
<tr>
<td>150</td>
<td>30</td>
<td>0.62</td>
<td>0.003</td>
<td>29.94</td>
<td>$145,750,032</td>
<td>150</td>
<td>$457,828,200</td>
<td>254,349,000</td>
</tr>
<tr>
<td>200</td>
<td>35</td>
<td>0.63</td>
<td>0.004</td>
<td>46.54</td>
<td>$142,683,919</td>
<td>200</td>
<td>$449,170,920</td>
<td>249,539,400</td>
</tr>
<tr>
<td>250</td>
<td>40</td>
<td>0.64</td>
<td>0.005</td>
<td>60.44</td>
<td>$138,878,062</td>
<td>250</td>
<td>$437,281,416</td>
<td>242,934,120</td>
</tr>
<tr>
<td>300</td>
<td>45</td>
<td>0.65</td>
<td>0.006</td>
<td>76.74</td>
<td>$147,740,511</td>
<td>300</td>
<td>$462,859,596</td>
<td>257,144,220</td>
</tr>
<tr>
<td>350</td>
<td>50</td>
<td>0.66</td>
<td>0.007</td>
<td>134.67</td>
<td>$146,658,068</td>
<td>350</td>
<td>$460,864,080</td>
<td>256,035,600</td>
</tr>
<tr>
<td>400</td>
<td>55</td>
<td>0.67</td>
<td>0.008</td>
<td>235.38</td>
<td>$142,268,314</td>
<td>400</td>
<td>$448,595,496</td>
<td>249,219,720</td>
</tr>
<tr>
<td>450</td>
<td>60</td>
<td>0.68</td>
<td>0.009</td>
<td>272.99</td>
<td>$145,935,209</td>
<td>450</td>
<td>$458,075,088</td>
<td>254,486,160</td>
</tr>
<tr>
<td>500</td>
<td>65</td>
<td>0.69</td>
<td>0.01</td>
<td>322.94</td>
<td>$142,030,433</td>
<td>500</td>
<td>$447,788,736</td>
<td>248,771,320</td>
</tr>
<tr>
<td>550</td>
<td>70</td>
<td>0.7</td>
<td>0.011</td>
<td>221.43</td>
<td>$145,014,922</td>
<td>550</td>
<td>$456,249,024</td>
<td>253,471,680</td>
</tr>
<tr>
<td>600</td>
<td>75</td>
<td>0.71</td>
<td>0.012</td>
<td>373.53</td>
<td>$147,756,261</td>
<td>600</td>
<td>$462,948,372</td>
<td>253,471,680</td>
</tr>
<tr>
<td>650</td>
<td>80</td>
<td>0.72</td>
<td>0.013</td>
<td>500.5</td>
<td>$147,283,315</td>
<td>650</td>
<td>$461,922,588</td>
<td>256,623,660</td>
</tr>
<tr>
<td>700</td>
<td>85</td>
<td>0.73</td>
<td>0.014</td>
<td>533.25</td>
<td>$146,907,122</td>
<td>700</td>
<td>$461,054,592</td>
<td>256,144,400</td>
</tr>
<tr>
<td>750</td>
<td>90</td>
<td>0.74</td>
<td>0.015</td>
<td>371.99</td>
<td>$146,860,378</td>
<td>750</td>
<td>$462,559,248</td>
<td>256,977,360</td>
</tr>
<tr>
<td>800</td>
<td>95</td>
<td>0.75</td>
<td>0.016</td>
<td>415.28</td>
<td>$147,404,600</td>
<td>800</td>
<td>$463,857,840</td>
<td>257,698,800</td>
</tr>
<tr>
<td>850</td>
<td>100</td>
<td>0.76</td>
<td>0.017</td>
<td>440.72</td>
<td>$145,015,461</td>
<td>850</td>
<td>$456,225,696</td>
<td>256,167,360</td>
</tr>
<tr>
<td>900</td>
<td>105</td>
<td>0.77</td>
<td>0.018</td>
<td>680.12</td>
<td>$144,136,531</td>
<td>900</td>
<td>$454,114,512</td>
<td>252,285,840</td>
</tr>
<tr>
<td>950</td>
<td>110</td>
<td>0.78</td>
<td>0.019</td>
<td>961.95</td>
<td>$144,791,458</td>
<td>950</td>
<td>$455,900,400</td>
<td>253,278,000</td>
</tr>
<tr>
<td>1000</td>
<td>115</td>
<td>0.79</td>
<td>0.02</td>
<td>604.45</td>
<td>$148,492,492</td>
<td>1000</td>
<td>$466,528,896</td>
<td>259,182,720</td>
</tr>
</tbody>
</table>
The best path were these results were obtained does not include any intermediate warehouse open. The harvesting units needed in order to harvest the 870,162 dry tons/year of switchgrass feedstock were 20. The total amount of residue generated in the biorefineries was 17,403.24 dry tons/year. The following figure represents the percentage of residue generated by month out of the total residue produced by year.

Table 7: BCO results: costs, revenue and profit

| Month | 1            | 2            | 3            | 4            | 5            | 6            | 7
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Processing Cost</td>
<td>$8,956,900.00</td>
<td>$4,887,300.00</td>
<td>$7,377,000.00</td>
<td>$8,938,200.00</td>
<td>$3,441,600.00</td>
<td>$10,206,700.00</td>
</tr>
<tr>
<td></td>
<td>Purchasing Cost</td>
<td>$8,956,900.00</td>
<td>$4,887,300.00</td>
<td>$5,163,900.00</td>
<td>$6,256,740.00</td>
<td>$2,409,120.00</td>
<td>$7,144,690.00</td>
</tr>
<tr>
<td></td>
<td>Inventory Cost</td>
<td>$243,582.76</td>
<td>$197,446.92</td>
<td>$242,950.80</td>
<td>$243,575.28</td>
<td>$139,040.64</td>
<td>$244,082.68</td>
</tr>
<tr>
<td></td>
<td>Transportation Truck</td>
<td>$716.55</td>
<td>$39,489.38</td>
<td>$59,606.16</td>
<td>$72,220.66</td>
<td>$275.33</td>
<td>$81,653.60</td>
</tr>
<tr>
<td></td>
<td>Transportation Train</td>
<td>$7,165.52</td>
<td>$0.00</td>
<td>$0.00</td>
<td>$0.00</td>
<td>$2,753.28</td>
<td>$81.65</td>
</tr>
<tr>
<td></td>
<td>Operating Cost</td>
<td>$1,726,666.67</td>
<td>$1,726,666.67</td>
<td>$1,726,666.67</td>
<td>$1,726,666.67</td>
<td>$1,726,666.67</td>
<td>$1,726,666.67</td>
</tr>
<tr>
<td></td>
<td>Harvest Unit</td>
<td>$966,666.67</td>
<td>$966,666.67</td>
<td>$966,666.67</td>
<td>$966,666.67</td>
<td>$966,666.67</td>
<td>$966,666.67</td>
</tr>
<tr>
<td></td>
<td>Revenue</td>
<td>$29,020,356.00</td>
<td>$15,834,852.00</td>
<td>$23,901,480.00</td>
<td>$28,959,768.00</td>
<td>$11,150,784.00</td>
<td>$33,069,708.00</td>
</tr>
<tr>
<td></td>
<td>Profit</td>
<td>$8,161,757.83</td>
<td>$3,129,982.36</td>
<td>$8,364,689.71</td>
<td>$10,755,698.73</td>
<td>$2,464,661.42</td>
<td>$12,699,166.73</td>
</tr>
</tbody>
</table>

Month 8 9 10 11 12 Total
Processing Cost | $10,960,500.00 | $4,544,500.00 | $8,545,500.00 | $3,295,200.00 | $11,819,200.00 | $87,016,200.00 |
Purchasing Cost | $10,960,500.00 | $4,544,500.00 | $8,545,500.00 | $3,295,200.00 | $11,819,200.00 | $78,027,150.00 |
Inventory Cost | $244,384.20 | $183,597.80 | $243,418.20 | $133,126.08 | $244,727.68 | $2,523,294.48 |
Transportation Truck | $88,560.84 | $36,719.56 | $69,047.64 | $26,361.60 | $95,499.14 | $602,822.74 |
Transportation Train | $0.00 | $0.00 | $0.00 | $26.36 | $0.00 | $10,026.82 |
Operating Cost | $1,726,666.67 | $1,726,666.67 | $1,726,666.67 | $1,726,666.67 | $1,726,666.67 | $5,20,000.00 |
Harvest Unit | $966,666.67 | $966,666.67 | $966,666.67 | $966,666.67 | $966,666.67 | $38,294,208.00 |
Revenue | $35,512,020.00 | $14,724,180.00 | $27,687,420.00 | $10,676,448.00 | $38,294,208.00 | $281,932,488.00 |
Profit | $10,564,741.63 | $2,721,529.31 | $7,590,620.83 | $1,233,200.63 | $11,622,247.85 | $81,432,993.96 |

Figure 26: BCO annual costs

The best path were these results were obtained does not include any intermediate warehouse open. The harvesting units needed in order to harvest the 870,162 dry tons/year of switchgrass feedstock were 20. The total amount of residue generated in the biorefineries was 17,403.24 dry tons/year. The following figure represents the percentage of residue generated by month out of the total residue produced by year.
6.4 BCO Sensitivity analysis

A sensitivity analysis was also performed in the BCO in order to determine what parameter values provide a better result to the logistic system design problem. The original parameters used in the algorithm were a number of 12 bees, the beta parameter, which controls the influence of the profitability of the fitness function, was equal to 10, and the alpha value, which controls the arcfitness variable, was equal to 1, the lambda value corresponded originally to .99, the number of iterations were 100, and the waggle dance scaling factor (K) was set to 10. Those parameter values were obtained from the research done by Wong et al 2008, 2009, were an application of BCO was shown to solve a Traveling Salesman Problem. The BCO algorithm is a method that was recently discover on the year 2001, as indicated in Chapter 4. Consequently, there are not a lot of investigations done according to what parameter values could be considered as the best options to solve a particular problem. However, according to the previous mentioned research, it was identified that the lambda value should be a number from 0 to 1 and also that the waggle dance scaling factor can be obtained by multiplying 10% by the number of iterations desired in the problem. Nevertheless, since there are not a lot of studies indicating between what range of values can be the number of bees, beta, and the alpha factors, it was necessary to do an
experiment in order to check different parameter values and see which of those values can help generate the best profit.

As indicated in the table below, the largest profit resulted when using a number of bees of 500, an alpha value of 25%, a beta amount of 3, a lambda value of 25%, and an iteration number of 1500. The profit obtained by using this parameter values was of $107,620,000, which is above the previous solution with $26,187,006. And the revenue resulted with the sensitivity analysis was of $372,595,589 more than the first revenue gained with $281,932,488. However, the time the computer spent in order to get this result was tedious and time consuming, with 3,891 seconds or approximately 1.08 hours. The computer used to run the program and do the sensitivity analysis was a third generation core i7 processor.

### Table 8: BCO sensitivity analysis

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Bees</th>
<th>Alpha</th>
<th>Beta</th>
<th>Lambda</th>
<th>Computational Time (S)</th>
<th>Best Profit</th>
<th>Iteration</th>
<th>Revenue</th>
<th>Biofuel (gal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>5</td>
<td>1</td>
<td>10</td>
<td>1</td>
<td>1.73</td>
<td>$101,614,346</td>
<td>1</td>
<td>$351,803,170</td>
<td>195,446,205.65</td>
</tr>
<tr>
<td>40</td>
<td>10</td>
<td>0.95</td>
<td>20</td>
<td>0.95</td>
<td>4.09</td>
<td>$103,198,434</td>
<td>15</td>
<td>$357,287,505</td>
<td>198,493,058.56</td>
</tr>
<tr>
<td>60</td>
<td>15</td>
<td>0.9</td>
<td>20</td>
<td>0.9</td>
<td>21.3</td>
<td>$104,496,004</td>
<td>1</td>
<td>$361,779,875</td>
<td>200,988,819.60</td>
</tr>
<tr>
<td>80</td>
<td>20</td>
<td>0.85</td>
<td>4</td>
<td>0.85</td>
<td>11.88</td>
<td>$103,064,614</td>
<td>48</td>
<td>$356,824,201</td>
<td>198,235,667.25</td>
</tr>
<tr>
<td>100</td>
<td>25</td>
<td>0.8</td>
<td>15</td>
<td>0.8</td>
<td>17.5</td>
<td>$104,917,694</td>
<td>90</td>
<td>$363,239,825</td>
<td>201,799,902.54</td>
</tr>
<tr>
<td>120</td>
<td>30</td>
<td>0.75</td>
<td>17</td>
<td>0.75</td>
<td>18.92</td>
<td>$104,537,713</td>
<td>65</td>
<td>$361,924,279</td>
<td>201,069,043.89</td>
</tr>
<tr>
<td>140</td>
<td>35</td>
<td>0.7</td>
<td>9</td>
<td>0.7</td>
<td>31.56</td>
<td>$104,767,235</td>
<td>93</td>
<td>$362,718,916</td>
<td>201,510,508.93</td>
</tr>
<tr>
<td>160</td>
<td>40</td>
<td>0.65</td>
<td>12</td>
<td>0.65</td>
<td>40.25</td>
<td>$103,817,212</td>
<td>64</td>
<td>$359,429,803</td>
<td>199,683,223.82</td>
</tr>
<tr>
<td>180</td>
<td>45</td>
<td>0.6</td>
<td>5</td>
<td>0.6</td>
<td>51.8</td>
<td>$104,090,024</td>
<td>138</td>
<td>$360,374,317</td>
<td>200,207,953.80</td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>0.55</td>
<td>18</td>
<td>0.55</td>
<td>54.37</td>
<td>$105,375,758</td>
<td>166</td>
<td>$364,825,706</td>
<td>202,680,947.57</td>
</tr>
<tr>
<td>220</td>
<td>60</td>
<td>0.5</td>
<td>6</td>
<td>0.5</td>
<td>77.88</td>
<td>$104,831,319</td>
<td>26</td>
<td>$362,940,782</td>
<td>201,633,767.56</td>
</tr>
<tr>
<td>500</td>
<td>70</td>
<td>1</td>
<td>20</td>
<td>1</td>
<td>177.89</td>
<td>$105,722,299</td>
<td>443</td>
<td>$366,025,481</td>
<td>203,347,489.31</td>
</tr>
<tr>
<td>1000</td>
<td>200</td>
<td>0.5</td>
<td>5</td>
<td>0.5</td>
<td>1148.32</td>
<td>$106,744,337</td>
<td>370</td>
<td>$369,563,921</td>
<td>205,313,289.37</td>
</tr>
<tr>
<td>1500</td>
<td>500</td>
<td>0.25</td>
<td>3</td>
<td>0.25</td>
<td>3981</td>
<td>$107,620,000</td>
<td>345</td>
<td>$372,595,589</td>
<td>206,997,549.41</td>
</tr>
</tbody>
</table>

### 6.5 GA and BCO Comparison

When both algorithms were adapted to the problem under study, the superior advantage that showed the GA over the performance of the BCO was noticeable. This bigger advantage is reflected in the results obtained when solving the same problem with the help of the Matlab software; both GA and BCO algorithms were coded in Matlab. The profit generated with the Genetic Algorithm was of an optimal profit of $135,404,783.58, and the BCO gave as a result a profit of $81,432,993.96 (Table 9). The profits mentioned above were obtained before the sensitivity analysis. Furthermore, if changes are
done to the parameters, better profits can be observed from both of the algorithms. However, even though both algorithms presented better results after the change in parameters, GA gave the optimal profit. Another important characteristic to specify is about the positive evolution that showed the GA because its best solutions were obtained during the last iteration contrary to the BCO whose best values were obtained from different iterations, but not the last one, such as the GA. However, in spite that BCO is below in the amount of profit with $53,971,789.62 than the GA, it still offers a good solution if there is a comparison with the results obtained in the model were this research is based. In the Mixed Linear Integer approach, the results from that investigation were an optimal profit of $60,000,000 million dollars (Zhu & Yao, 2011). Nevertheless, it is important to declare that the model was not followed identically, due to it was necessary to do some changes in order to apply the two metaheuristics. Even though there were changes made, this experiment demonstrates that both algorithms can be developed and adapted to the logistic system design problem including three different types of biomass feedstock.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Mixed Integer Linear Programming</th>
<th>Genetic Algorithm (proposed)</th>
<th>Bee Colony Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profit</td>
<td>$60,000,000</td>
<td>$135,404,783.58</td>
<td>$81,432,993.96</td>
</tr>
</tbody>
</table>

### 6.6 Summary

Chapter 6 presents the results obtained from the numerical example about the logistic system design problem. Furthermore, the computation results of the problem are displayed and explained. The solution showed better results with the GA than with the BCO algorithm. Furthermore, a description of the sensitivity analysis performed in order to test different parameter values for the two metaheuristic was done.
CHAPTER 7: CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

7.1 General

At the present time the nation must find other sources of energy besides fossil fuels. The consistent burning of fossil fuels has caused damage to the environment. Another problem that must be taken into account is that the resources of fossil fuels that the planet has are finite. That is why it is vital for the nation to consider the use of alternative sources of energy. One good example of this alternative energy source is biofuel produced from biomass. However, before biofuels can be considered as a stable alternative form of fuel, some logistical challenges must be solved. Several studies have been made trying to solve those logistical challenges; this was done with several optimization techniques. To the best of my knowledge, metaheuristics algorithms were not used before to solve this type of problems. That is why in this thesis it was decided to use metaheuristics algorithms to show that they can provide better solutions than other mathematical methods.

The use of multi-commodity systems can be very beneficial to minimize the huge costs caused by producing biofuels. But at the present time, there are still a lot of logistic challenges, such as the difficulty in the availability of the biomass feedstock according to the season of the year, the need of special equipment to collect the large amounts of feedstock, the work force require to meet the harvesting, the necessity in the increase of transportation due to the low density of the biomass, having the proper storage space to not lose the properties of the biomass when it is produced, and the necessity of having some customize machinery or equipment in order to process a different type of biomass, which may be needed to be treated differently, etc…. These are just some of the logistic challenges which are not permitting the increase in the production of renewable energy produced by the biomass feedstock. The implementation of energy produced by biomass is vital for the country in order to avoid
the great dependency that the U.S. has on other foreign countries in order to supply the oil demanded in the country. And according to recent investigations, the biomass feedstock is the only green source that can replace the energy produced by the oil. It is well-known that in order to produce biofuels from any type of biomass, it will incur in the release of contaminants to the atmosphere. In other words, they are not totally green; however, they will help to lessen the negative impact of the use of fossil fuels, such as petroleum.

7.2 Conclusions

The conclusions of this thesis are the following:

- GA resulted in the best profit with $135,404,783.58 and revenue of $427,159,332.00.
- BCO did not provide the best profit ($81,432,993.96), but still gives a better result than the MILP method ($60,000,000).
- The largest annual costs for both GA and BCO were processing and purchasing costs; GA’s processing cost was $131,839,300 while its purchasing cost was $118,135,570, and for BCO processing cost was $ 87,016,200 while BCO’s purchasing cost was $78,027,150.
- After performing the sensitivity analysis GA provided again the best profit with $148,492,492 and BCO resulted in a profit of $107,620,000.
- The computation time in order to get the best solutions from the sensitivity analysis was approximately 10 minutes for GA and 1.08 hours for BCO.

7.3 Contributions

This thesis contributes to the biomass logistic system design problem by:

1. Introducing the GA and BCO algorithms.
2. Showing that good results can be obtained from the proposed metaheuristics in this area.
3. Demonstrates that GA and BCO algorithms outperform the achievement of the MILP.
4. Providing new insights to the industrial engineering community about the approach showed in the methodology.

7.4 Recommendations for Future Work

There are still many challenges to analyze in the logistics of biomass to biofuel supply chain. It would be an interesting future work to explore other optimization algorithms, such as Particle Swarm Optimization (PSO) or Simulated Annealing (SA), for logistic system design problem and perform multi-objective optimization problems as well. Another thing that can be done is to solve the same problem in a more dynamic way. This could be done by including probabilistic techniques, for instance Markov Chains, in order to add a more realistic scenario so that the system could be more robust.
REFERENCES


APPENDIX A

Matlab Code for Genetic Algorithm
tic
clear
clc
popsize=105;
generations=900;
perc=.77;
mutation=.018;
transportation=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 0 0 0 0];
x=1;
while x<=popsize
l=1;
  %generar meses
  months_short=randperm(12);
  s=size(months_short,2);
  %generar tipo de biomass
  for i=1:s
    if
      months_short(1,i)==1||months_short(1,i)==2||months_short(1,i)==7||months_short(1,i)==8||months_short(1,i)==9||months_sh
      ort(1,i)==10||months_short(1,i)==11||months_short(1,i)==12
      btype(1,i)=1;
else
    btype(1,i)=2+1.*round(rand(1,1));
end
end
j=1;

%% generar field of origin
m2=randperm(10);
ttt=1;
%m3=10.+randperm(2)
%m4=12.+randperm(2)
while j<=s
    if btype(1,j)==1
        field(1,j)=m2(ttt);
        j=1+size(field,2);
        ttt=ttt+1;
    elseif btype(1,j)==2
        field(1,j)=11+1.*round(rand(1,1));
        j=1+size(field,2);
    else
        field(1,j)=13+1.*round(rand(1,1));
        j=1+size(field,2);
    end
end
field_short(:,1:s)=field(1,1:s);

%% generar # of tons processed
for k=1:s
    tons(1,k)=30000+round(90000*rand(1,1));
end

%% generar biorefineries
bioref(:,1:s)=3.*ones(1,1:s);

%% tons of residue
for o=1:s
    if bioref(1,o)==3
        tons_res(1,o)=2*0.01*tons(1,o);
    else
        tons_res(1,o)=0.01*tons(1,o);
    end
end
sumas=sum(tons_res,2);
%% residue recirculated to field
for p=1:s
    field_recirc(1,p)=1+round(9*rand(1,1));
end

%% in-field warehouse
infwareh(:,1:s)=field_recirc(1,1:s);

%% # of tons of residue stored
tons_res_st(:,1:s)=tons_res(:,1:s);

%% type of biomass stored
btype_stored(:,1:s)=2.*ones(1,1:s);

%% type of biomass transported
btype_transp(:,1:s)=btype(1,1:s);

%% from-field
field_from(:,1:s)=field_short(1,1:s);

%% intermediate warehouse open
for q=1:s
    if field_from(1,q)==11||field_from(1,q)==12||field_from(1,q)==13||field_from(1,q)==14
        int_wareh_open(1,q)=0;
    else
        int_wareh_open(1,q)=round(rand(1,1));
    end
end

%% intermediate warehouse
for r=1:s
    if int_wareh_open(1,r)==0
        int_wareh(1,r)=0;
    else
        int_wareh(1,r)=15+round(2*rand(1,1));
    end
end

%% transportation type
for t=1:s
    if int_wareh(1,t)==0
        trans_type(1,t)=0;
    else
        trans_type(1,t)=transportation(field_from(1,t),int_wareh(1,t));
    end
end

%% to biorefinery
for u=1:s
    if bioref(1,u)==1
        bio_to(1,u)=18;
    elseif bioref(1,u)==2
        bio_to(1,u)=19;
    else
        bio_to(1,u)=20;
    end
end
%% transportation type
for v=1:s
    if int_wareh_open(1,v)==0
        trans_type_1(1,v)=transportation(field_from(1,v),bio_to(1,v));
    else
        trans_type_1(1,v)=transportation(int_wareh(1,v),bio_to(1,v));
    end
end
%% # of tons transported
tons_trans(:,1:s)=tons(1,1:s);
%% residue transported
resi_trans(1,1:s)=4*ones(1,1:s);
%% residue from
resi_from(1,1:s)=bio_to(1,1:s);
%% to field
resi_to(1,1:s)=infwareh(1,1:s);
%% residue transportation type
for w=1:s
    resi_trans_type(1,w)=transportation(resi_from(1,w),resi_to(1,w));
end
%% # of tons of residue transported
tons_resi_trans(1,1:s)=tons_res_st(1,1:s);

chr=[months_short; btype; field_short; tons; bioref; tons_res; field_recir; infwareh; tons_res_st; btype_stored; btype_transp; field_from; int_wareh_open; int_wareh; trans_type; bio_to; trans_type_1; tons_trans; resi_trans; resi_from; resi_to; resi_trans_type; tons_resi_trans];
%% constraint of minimum and maximum of tons stored at biorefineries
suma=sum(chr(4,:),2)*2;
if suma(1,1)>2100000
    constraint1(1,1)=0;
end
else
    constraint1(1,1)=1;
end

if constraint1(1,1)==0
    clearvars bio_to bioref btype btype_stored btype_transp chr chr1 field field_from field_recir field_short
    clearvars i infwareh int_wareh int_wareh_open j k l m m1 m2 month months months_short n o p q r resi_from
    resi_to
    clearvars resi_trans resi_trans_type t tons tons_res tons_res_st tons_resi_trans tons_trans trans_type
trans_type_1
    clearvars u v w y z is1 aa bb sum_tons sum_tons_final suma constraint1 cc switch_prod
    continue
else
    chr1=[months_short , btype , field_short , tons , bioref , tons_res , field_recir , infwareh , tons_res_st , btype_stored,
btype_transp , field_from , int_wareh_open , int_wareh , trans_type , bio_to , trans_type_1 , tons_trans , resi_trans , resi_from,
resi_to , resi_trans_type , tons_resi_trans];

    if x==1
        chrom1(:, :) = chr(:, :);
    elseif x==2
        chrom2(:, :) = chr(:, :);
    else
        chrom3(:, :) = chr(:, :);
    end

    population(x, :) = chr1(:, :);
end

if size(population, 1) <= popsize
    x = x + 1;
else
    x = x;
end
%%processing cost
for dd=1:popsize
    processing_cost(dd,1)=50*sum(population(dd,37:48))*2;
end

%%feedstock purchasing cost
for ee=1:popsize
    for ff=1:s
        if population(ee,(ff+s))==1
            switchgrass(1,ff)=population(ee,(ff+36));
            stalk_straw(1,ff)=0;
        else
            switchgrass(1,ff)=0;
            stalk_straw(1,ff)=population(ee,(ff+36));
        end
    end
    feed_purch_cost(ee,1)=50*sum(switchgrass)*2+35*sum(stalk_straw)*2;
clearvars ff switchgrass stalk_straw
end

%%inventory cost
for gg=1:popsize
    residue_sum(gg,1)=sum(population(gg,61:72));
    for hh=1:s
        if population(gg,(hh+144))==1
            int_warehouse(1,hh)=population(gg,(hh+36));
        else
            int_warehouse(1,hh)=0;
        end
    end
    int_warehouse_sum(gg,1)=sum(int_warehouse);
    for ii=1:s
        is2=ismember(population(gg,1:s),ii);
        for jj=1:s
            if is2(1,jj)==0
                sum_tons1(1,jj)=0;
            else
                sum_tons1(1,jj)=population(gg,jj+36);
            end
        end
    end
end
sum_tons1(sum_tons1==0)=[];
if isempty(sum_tons1)==1
    sum_tons_final1(ii,1)=0;
else
    sum_tons_final1(ii,1:size(sum_tons1,2))=sum_tons1(1,:);
end
end
suma1(1:s,1)=sum(sum_tons_final1,2);
suma1=suma1';
for kk=1:12
    if suma1(1,kk)>60000
        in_bio(1,kk)=60000;
    else
        in_bio(1,kk)=suma1(1,kk);
    end
end
in_bio_sum(gg,1)=sum(in_bio);
inventory_cost(gg,1)=2*(residue_sum(gg,1)+int_warehouse_sum(gg,1)*2+in_bio_sum(gg,1)*2);
end
%%transportation cost by trucks
for ll=1:1:size(population,1)
    for mm=1:size(population,2)
        if population(ll,(mm+168))==1
            transported1(1,mm)=population(ll,(mm+204));
        else
            transported1(1,mm)=0;
        end
        if population(ll,(mm+192))==1
            transported2(1,mm)=population(ll,(mm+204));
        else
            transported2(1,mm)=0;
        end
        if population(ll,(mm+252))==1
            transported3(1,mm)=population(ll,(mm+264));
        else
            transported3(1,mm)=0;
        end
    end
end
truck_transport(ll,1)=.4*(sum(transported1)*2+sum(transported2)*2+sum(transported3));

%%transportation cost by train
for nn=1:popsize
    for oo=1:s
        if population(nn,(oo+168))==2
            transported4(1,oo)=population(nn,(oo+204));
        else
            transported4(1,oo)=0;
        end
        if population(nn,(oo+192))==2
            transported5(1,oo)=population(nn,(oo+204));
        else
            transported5(1,oo)=0;
        end
        if population(nn,(oo+252))==2
            transported6(1,oo)=population(nn,(oo+264));
        else
            transported6(1,oo)=0;
        end
    end
    train_transport(nn,1)=.04*(sum(transported4)*2+sum(transported5)*2+sum(transported6));
end

%%operation cost of warehouses and biorefineries
for pp=1:popsize
    location=find(population(pp,145:156));
    months_int_wareh=population(pp,(location));
    months_intwarehouse=unique(months_int_wareh);
    size=size(months_intwarehouse,2);
    operation_cost_wb(pp,1)=12*30000*2+10000000*2+60000*size;
    clearvars location months_int_wareh months_intwarehouse size
end

%%operation cost for harvest units
for qq=1:popsize
    for rr=1:s
        if population(qq,(rr+s))==1
            switchgrass1(1,rr)=population(qq,(rr+36));
        else
            switchgrass1(1,rr)=0;
        end
    end
end
total_switchgrass = sum(switchgrass1) * 2;
number_units(qq,1) = total_switchgrass / 8 / 7200;
operation_cost(qq,1) = 580000 * ceil(number_units(qq,1));

%% revenue
for ss = 1:popsize
    total_processed = sum(population(ss,37:48)) * 2;
    revenue(ss,1) = 1.8 * 90 * total_processed;
end

%% profit
for tt = 1:popsize
    profit(tt,1) = revenue(tt,1) - processing_cost(tt,1) - feed_purch_cost(tt,1) - inventory_cost(tt,1) - truck_transport(tt,1) - train_transport(tt,1) - operation_cost_wb(tt,1) - operation_cost(tt,1);
end

%% rank selection
[profit_sort,order] = sort(profit, 'descend');
pop2(:, :) = population(order,:);
rank_perc = round(perc * popsize);
red_pop(1:rank_perc,:) = pop2(1:rank_perc,:);

%% single point crossover
counter = 1;
for uu = 1:rank_perc-1
    child1_1(1,1:36) = red_pop(uu,1:36);
    child1_2(1,1:12) = [red_pop(uu,37:42) red_pop(uu+1,43:48)];
    child1_3(1,1:12) = red_pop(uu,49:60);
    for ww = 1:s
        child1_4(1,ww) = 2 * 0.01 * child1_2(1,ww);
    end
    child1_5(1,1:12) = [red_pop(uu,73:78) red_pop(uu+1,79:84)];
    child1_6(1,1:12) = child1_5(1,1:12);
    child1_7(1,1:12) = child1_4(1,1:12);
    child1_8(1,1:36) = red_pop(uu,109:144);
    child1_9(1,1:12) = [red_pop(uu,145:150) red_pop(uu+1,151:156)];
    for xx = 1:s
        if child1_9(1,xx) == 0
            child1_10(1,xx) = 0;
        end
    end
end
else
    child1_10(1,xx)=15+round(2*rand(1,1));
end
end
for yy=1:s
    if child1_10(1,yy)==0
        child1_11(1,yy)=0;
    else
        child1_11(1,yy)=transportation(child1_8(1,yy+24),child1_10(1,yy));
    end
end
child1_12(1,1:12)=red_pop(uu,181:192);
for zz=1:s
    if child1_9(1,zz)==0
        child1_13(1,zz)=transportation(child1_8(1,zz+24),child1_12(1,zz));
    else
        child1_13(1,zz)=transportation(child1_10(1,zz),child1_12(1,zz));
    end
end
child1_14(1,1:12)=child1_2(1,1:12);
child1_15(1,1:24)=red_pop(uu,217:240);
child1_16(1,1:12)=child1_6(1,1:12);
for aaa=1:s
    child1_17(1,aaa)=transportation(child1_15(1,aaa+12),child1_16(1,aaa));
end
child1_18(1,1:12)=child1_7(1,1:12);
child1=[child1_1 child1_2 child1_3 child1_4 child1_5 child1_6 child1_7 child1_8 child1_9 child1_10 child1_11 child1_12 child1_13 child1_14 child1_15 child1_16 child1_17 child1_18];

% child 2
child2_1(1,1:36)=red_pop(uu+1,1:36);
child2_2(1,1:12)=[red_pop(uu+1,37:42) red_pop(uu,43:48)];
child2_3(1,1:12)=red_pop(uu+1,49:60);
for ww=1:s
    child2_4(1,ww)=2*0.01*child2_2(1,ww);
end
child2_5(1,1:12)=[red_pop(uu+1,73:78) red_pop(uu,79:84)];
child2_6(1,1:12)=child2_5(1,1:12);
child2_7(1,1:12)=child2_4(1,1:12);
child2_8(1,1:36)=red_pop(uu+1,109:144);
child2_9(1,1:12)=[red_pop(uu+1,145:150) red_pop(uu,151:156)];
for xx=1:s
    if child2_9(1,xx)==0
        child2_10(1,xx)=0;
    else
        child2_10(1,xx)=15+round(2*rand(1,1));
    end
end
for yy=1:s
    if child2_10(1,yy)==0
        child2_11(1,yy)=0;
    else
        child2_11(1,yy)=transportation(child2_8(1,yy+24),child2_10(1,yy));
    end
end
child2_12(1,1:12)=red_pop(uu+1,181:192);
for zz=1:s
    if child2_9(1,zz)==0
        child2_13(1,zz)=transportation(child2_8(1,zz+24),child2_12(1,zz));
    else
        child2_13(1,zz)=transportation(child2_10(1,zz),child2_12(1,zz));
    end
end
child2_14(1,1:12)=child2_2(1,1:12);
child2_15(1,1:24)=red_pop(uu+1,217:240);
child2_16(1,1:12)=child2_6(1,1:12);
for aaa=1:s
    child2_17(1,aaa)=transportation(child2_15(1,aaa+12),child2_16(1,aaa));
end
child2_18(1,1:12)=child2_7(1,1:12);
child2=[child2_1 child2_2 child2_3 child2_4 child2_5 child2_6 child2_7 child2_8 child2_9 child2_10 child2_11
child2_12 child2_13 child2_14 child2_15 child2_16 child2_17 child2_18];

children([counter counter+1,:]=[child1;child2];
counter=counter+2;
end

child=[children(1,1:12);children(1,13:24);children(1,25:36);children(1,37:48);children(1,49:60);children(1,61:72);c
hildren(1,73:84);children(1,85:96);children(1,97:108);children(1,109:120);children(1,121:132);children(1,133:144);children(1,145:150);children(1,151:156)];

children([counter counter+1,:]=children;
children_red(1:popsize,:)=children(1:popsize,:);

mutation_perc=ceil(mutation*size(children_red,1));
if mutation_perc>1
  row_mutation(1,1)=1+round((popsize-1)*rand(1,1));
else
  row_mutation(1,size(mutation_perc,2))=1+round((popsize-1)*rand(1,mutation_perc));
end
places=randperm(12);

for bbb=1:size(row_mutation,2)
  mutated_child(1,:)=children_red(row_mutation(1,bbb,:));
  mutatedchild1_1(1,1:36)=mutated_child(1,1:36);
  places_switch(1,1:2)=places(1,1:2);
  mutatedchild1_2(1,1:12)=mutated_child(1,37:48);
  mutatedchild1_2(1,[places_switch(1,1) places_switch(1,2)])=mutatedchild1_2(1,[places_switch(1,2) places_switch(1,1)]);
  mutatedchild1_3(1,1:12)=mutated_child(1,49:60);
  for ccc=1:s
    mutatedchild1_4(1,ccc)=2*0.01*mutatedchild1_2(1,ccc);
  end
  mutatedchild1_5(1,1:12)=mutated_child(1,73:84);
  mutatedchild1_5(1,[places_switch(1,1) places_switch(1,2)])=mutatedchild1_5(1,[places_switch(1,2) places_switch(1,1)]);
  mutatedchild1_6(1,1:12)=mutatedchild1_5(1,1:12);
  mutatedchild1_7(1,1:12)=mutatedchild1_4(1,1:12);
  mutatedchild1_8(1,1:36)=mutated_child(1,109:144);
  mutatedchild1_9(1,1:12)=mutated_child(1,145:156);
  mutatedchild1_9(1,[places_switch(1,1) places_switch(1,2)])=mutatedchild1_9(1,[places_switch(1,2) places_switch(1,1)]);
  for ddd=1:s
    if mutatedchild1_9(1,ddd)==0
      mutatedchild1_10(1,ddd)=0;
    else
      mutatedchild1_10(1,ddd)=15+round(2*rand(1,1));
    end
  end
end
for eee=1:s
    if mutatedchild1_10(1,eee)==0
        mutatedchild1_11(1,eee)=0;
    else
        mutatedchild1_11(1,eee)=transportation(mutatedchild1_8(1,eee+24),mutatedchild1_10(1,eee));
    end
end
mutatedchild1_12(1,1:12)=mutated_child(1,181:192);
for fff=1:s
    if mutatedchild1_9(1,fff)==0
        mutatedchild1_13(1,fff)=transportation(mutatedchild1_8(1,fff+24),mutatedchild1_12(1,fff));
    else
        mutatedchild1_13(1,fff)=transportation(mutatedchild1_10(1,fff),mutatedchild1_12(1,fff));
    end
end
mutatedchild1_14(1,1:12)=mutatedchild1_2(1,1:12);
mutatedchild1_15(1,1:24)=mutated_child(1,217:240);
mutatedchild1_16(1,1:12)=mutatedchild1_6(1,1:12);
for ggg=1:s
    mutatedchild1_17(1,ggg)=transportation(mutatedchild1_15(1,ggg+12),mutatedchild1_16(1,ggg));
end
mutatedchild1_18(1,1:12)=mutatedchild1_7(1,1:12);
mutatedchild1=[mutatedchild1_1 mutatedchild1_2 mutatedchild1_3 mutatedchild1_4 mutatedchild1_5
mutatedchild1_6 mutatedchild1_7 mutatedchild1_8 mutatedchild1_9 mutatedchild1_10 mutatedchild1_11 mutatedchild1_12
mutatedchild1_13 mutatedchild1_14 mutatedchild1_15 mutatedchild1_16 mutatedchild1_17 mutatedchild1_18];
end
children_red(row_mutation(1,1,:),:)=mutatedchild1(1,:);
best_profit1=profit_sort(1,1);
clearvars aaa bbb ccc child child1 child1_1 child1_10 child1_11 child1_12 child1_13 child1_14 child1_15
child1_16 child1_17 child1_18 child1_2 child1_3 child1_4 child1_5 child1_6 child1_7
clearvars child1_8 child1_9 child2 child2_1 child2_2 child2_3 child2_4 child2_5 child2_6 child2_7
clearvars child2_8 child2_9 children chrom1 chrom2 chrom3 counter dd ddd ee eee feed_purch_cost fff ggg hh ii
in_bio in_bio_sum int_warehouse int_warehouse_sum inventory_cost is2 jj kk
clearvars ll mm mutated_child mutatedchild1 mutatedchild1_1 mutatedchild1_11 mutatedchild1_12
mutatedchild1_13 mutatedchild1_14 mutatedchild1_15 mutatedchild1_16 mutatedchild1_17 mutatedchild1_18

93
clearvars mutatedchild1_2 mutatedchild1_3 mutatedchild1_4 mutatedchild1_5 mutatedchild1_6 mutatedchild1_7 mutatedchild1_8 mutatedchild1_9 mutation_perc nn number_units oo operation_cost operation_cost_wb

clearvars order places places_switch pop2 population pp processing_cost profit profit_sort qq rank_perc red_pop residue_sum revenue row_mutation rr ss sum_tons1 sum_tons_final1 sumal1 switchgrass1

clearvars total_processed total_switchgrass train_transport transported1 transported2 transported3 transported4 transported5 transported6 truck_transport tt uu ww xx yy zz

%%End first iteration

for hhh=1:generations-1

    %%processing cost
    for dd=1:popsize
        processing_cost(dd,1)=50*sum(children_red(dd,37:48))*2;
    end

    %%feedstock purchasing cost
    for ee=1:popsize
        for ff=1:s
            if children_red(ee,(ff+s))==1
                switchgrass(1,ff)=children_red(ee,(ff+36));
                stalk_straw(1,ff)=0;
            else
                switchgrass(1,ff)=0;
                stalk_straw(1,ff)=children_red(ee,(ff+36));
            end
        end
        feed_purch_cost(ee,1)=50*sum(switchgrass)*2+35*sum(stalk_straw)*2;
        clearvars ff switchgrass stalk_straw
    end

    %%inventory cost
    for gg=1:popsize
        residue_sum(gg,1)=sum(children_red(gg,61:72));
        for hh=1:s
            if children_red(gg,(hh+144))==1
                int_warehouse(1,hh)=children_red(gg,(hh+36));
            else
                int_warehouse(1,hh)=0;
            end
        end
    end
int_warehouse_sum(gg,1)=sum(int_warehouse);
for ii=1:s
    is2=ismember(children_red(gg,1:s),ii);
    for jj=1:s
        if is2(1,jj)==0
            sum_tons1(1,jj)=0;
        else
            sum_tons1(1,jj)=children_red(gg,jj+36);
        end
    end
    sum_tons1(sum_tons1==0)=[];
    if isempty(sum_tons1)==1
        sum_tons_final1(ii,1)=0;
    else
        sum_tons_final1(ii,1:size(sum_tons1,2))=sum_tons1(1,:);
    end
end
suma1(1:s,1)=sum(sum_tons_final1,2);
suma1=suma1';
for kk=1:12
    if suma1(1,kk)>60000
        in_bio(1,kk)=60000;
    else
        in_bio(1,kk)=suma1(1,kk);
    end
end
in_bio_sum(gg,1)=sum(in_bio);
inventory_cost(gg,1)=2*(residue_sum(gg,1)+int_warehouse_sum(gg,1)*2+in_bio_sum(gg,1)*2);
end
%%transportation cost by trucks
for ll=1:popsize
    for mm=1:s
        if children_red(ll,(mm+168))==1
            transported1(1,mm)=children_red(ll,(mm+204));
        else
            transported1(1,mm)=0;
        end
        if children_red(ll,(mm+192))==1
            transported1(1,mm)=children_red(ll,(mm+204));
        end
    end
end
transported2(1, mm) = children_red(ll, (mm + 204));
else
    transported2(1, mm) = 0;
end
if children_red(ll, (mm + 252)) = 1
    transported3(1, mm) = children_red(ll, (mm + 264));
else
    transported3(1, mm) = 0;
end
end

truck_transport(ll, 1) = 0.4 * (sum(transported1) * 2 + sum(transported2) * 2 + sum(transported3));
end

%% transportation cost by train
for nn = 1:popsize
    for oo = 1:s
        if children_red(nn, (oo + 168)) = 2
            transported4(1, oo) = children_red(nn, (oo + 204));
        else
            transported4(1, oo) = 0;
        end
        if children_red(nn, (oo + 192)) = 2
            transported5(1, oo) = children_red(nn, (oo + 204));
        else
            transported5(1, oo) = 0;
        end
        if children_red(nn, (oo + 252)) = 2
            transported6(1, oo) = children_red(nn, (oo + 264));
        else
            transported6(1, oo) = 0;
        end
    end
    train_transport(nn, 1) = 0.04 * (sum(transported4) * 2 + sum(transported5) * 2 + sum(transported6));
end

%% operation cost of warehouses and biorefineries
for pp = 1:popsize
    location = find(children_red(pp, 145:156));
    months_int_wareh = children_red(pp, (location));
    months_intwarehouse = unique(months_int_wareh);
    ise = isempty(location);
end
if ise==1
    size=0;
else
    size=size(months_intwarehouse,2);
end
operation_cost_wb(pp,1)=12*30000*2+10000000*2+60000*size;
clearvars location months_int_wareh months_intwarehouse size
end
%%operation cost for harvest units
for qq=1:popsize
    for rr=1:s
        if children_red(qq,(rr+s))==1
            switchgrass1(1,rr)=children_red(qq,(rr+36));
        else
            switchgrass1(1,rr)=0;
        end
    end
    total_switchgrass=sum(switchgrass1)*2;
    number_units(qq,1)=total_switchgrass/87200;
    operation_cost(qq,1)=580000*ceil(number_units(qq,1));
end
%%revenue
for ss=1:popsize
    total_processed=sum(children_red(ss,37:48))*2;
    revenue(ss,1)=1.8*90*total_processed;
end
%%profit
for tt=1:popsize
    profit(tt,1)=revenue(tt,1)-processing_cost(tt,1)-feed_purch_cost(tt,1)-inventory_cost(tt,1)-truck_transport(tt,1)-train_transport(tt,1)-operation_cost_wb(tt,1)-operation_cost(tt,1);
end
%%rank selection
[profit_sort,order]=sort(profit, 'descend')
pop2(:,:)=children_red(order,:);
rank_perc=round(perc*popsize);
red_pop(1:rank_perc,:)=pop2(1:rank_perc,:);

%%%single point crossover
counter=1;
for uu=1:rank_perc-1
    child1_1(1,1:36)=red_pop(uu,1:36);
    child1_2(1,1:12)=[red_pop(uu,37:42) red_pop(uu+1,43:48)];
    child1_3(1,1:12)=red_pop(uu,49:60);
    for ww=1:s
        child1_4(1,ww)=2*0.01*child1_2(1,ww);
    end
    child1_5(1,1:12)=[red_pop(uu,73:78) red_pop(uu+1,79:84)];
    child1_6(1,1:12)=child1_5(1,1:12);
    child1_7(1,1:12)=child1_4(1,1:12);
    child1_8(1,1:36)=red_pop(uu,109:144);
    child1_9(1,1:12)=[red_pop(uu,145:150) red_pop(uu+1,151:156)];
    for xx=1:s
        if child1_9(1,xx)==0
            child1_10(1,xx)=0;
        else
            child1_10(1,xx)=15+round(2*rand(1,1));
        end
    end
    for yy=1:s
        if child1_10(1,yy)==0
            child1_11(1,yy)=0;
        else
            child1_11(1,yy)=transportation(child1_8(1,yy+24),child1_10(1,yy));
        end
    end
    child1_12(1,1:12)=red_pop(uu,181:192);
    for zz=1:s
        if child1_9(1,zz)==0
            child1_13(1,zz)=transportation(child1_8(1,zz+24),child1_12(1,zz));
        else
            child1_13(1,zz)=transportation(child1_10(1,zz),child1_12(1,zz));
        end
    end
    child1_14(1,1:12)=child1_2(1,1:12);
    child1_15(1,1:24)=red_pop(uu,217:240);
    child1_16(1,1:12)=child1_6(1,1:12);
    for aaa=1:s
        child1_17(1,aaa)=transportation(child1_15(1,aaa+12),child1_16(1,aaa));
end
child1_18(1,1:12)=child1_7(1,1:12);
child1=[child1_1 child1_2 child1_3 child1_4 child1_5 child1_6 child1_7 child1_8 child1_9 child1_10 child1_11 child1_12 child1_13 child1_14 child1_15 child1_16 child1_17 child1_18];

%child 2
child2_1(1,1:36)=red_pop(uu+1,1:36);
child2_2(1,1:12)=[red_pop(uu+1,37:42) red_pop(uu,43:48)];
child2_3(1,1:12)=red_pop(uu+1,49:60);
for ww=1:s
    child2_4(1,ww)=2*0.01*child2_2(1,ww);
end
child2_5(1,1:12)=[red_pop(uu+1,73:78) red_pop(uu,79:84)];
child2_6(1,1:12)=child2_5(1,1:12);
child2_7(1,1:12)=child2_4(1,1:12);
child2_8(1,1:36)=red_pop(uu+1,109:144);
child2_9(1,1:12)=[red_pop(uu+1,145:150) red_pop(uu,151:156)];
for xx=1:s
    if child2_9(1,xx)==0
        child2_10(1,xx)=0;
    else
        child2_10(1,xx)=15+round(2*rand(1,1));
    end
end
for yy=1:s
    if child2_10(1,yy)==0
        child2_11(1,yy)=0;
    else
        child2_11(1,yy)=transportation(child2_8(1,yy+24),child2_10(1,yy));
    end
end
child2_12(1,1:12)=red_pop(uu+1,181:192);
for zz=1:s
    if child2_9(1,zz)==0
        child2_13(1,zz)=transportation(child2_8(1,zz+24),child2_12(1,zz));
    else
        child2_13(1,zz)=transportation(child2_10(1,zz),child2_12(1,zz));
    end
end
end
child2_{14}(1,1:12)=child2_{2}(1,1:12);
child2_{15}(1,1:24)=red_pop(uu+1,1,1:24);
child2_{16}(1,1:12)=child2_{6}(1,1:12);
for aaa=1:s
    child2_{17}(1,aaa)=transportation(child2_{15}(1,aaa+12),child2_{16}(1,aaa));
end
child2_{18}(1,1:12)=child2_{7}(1,1:12);
child2=[child2_{1} child2_{2} child2_{3} child2_{4} child2_{5} child2_{6} child2_{7} child2_{8} child2_{9} child2_{10}
child2_{11} child2_{12} child2_{13} child2_{14} child2_{15} child2_{16} child2_{17} child2_{18}];

children([counter counter+1],:)=child1;child2;
counter=counter+2;
end
clearvars children_red

cild=[children(1,1:12);children(1,13:24);children(1,25:36);children(1,37:48);children(1,49:60);children(1,61:72);children(1,
73:84);children(1,85:96);children(1,97:108);children(1,109:120);children(1,121:132);children(1,133:144);children(1,145:156
);children(1,157:168);children(1,169:180);children(1,181:192);children(1,193:204);children(1,205:216);children(1,217:228);
children(1,229:240);children(1,241:252);children(1,253:264);children(1,265:276)];
children_red(1:popsize,:)=child1;popsize,:);

mutation_perc=ceil(mutation*size(children_red,1));
if mutation_perc>1
    row_mutation(1,1)=1+round((popsize-1)*rand(1,1));
else
    row_mutation(1,size(mutation_perc,2))=1+round((popsize-1)*rand(1,mutation_perc));
end
places=randperm(12);

for bbb=1:size(row_mutation,2)
    mutated_child(1,:)=children_red(row_mutation(1,bbb),:);
    mutatedchild1_1(1,1:36)=mutated_child(1,1:36);
    places_switch(1,1:2)=places(1,1:2);
    mutatedchild1_2(1,1:12)=mutated_child(1,37:48);
    mutatedchild1_2(1,[places_switch(1,1) places_switch(1,2)])=mutatedchild1_2(1,[places_switch(1,2)
places_switch(1,1)]);
    mutatedchild1_3(1,1:12)=mutated_child(1,49:60);
    for ccc=1:s
        mutatedchild1_4(1,ccc)=2*0.01*mutatedchild1_2(1,ccc);
end
mutatedchild1_5(1,1:12)=mutated_child(1,73:84);
mutatedchild1_5(1,[places_switch(1,1), places_switch(1,2)])=mutatedchild1_5(1,[places_switch(1,2), places_switch(1,1)]);
mutatedchild1_6(1,1:12)=mutatedchild1_5(1,1:12);
mutatedchild1_7(1,1:12)=mutatedchild1_4(1,1:12);
mutatedchild1_8(1,1:36)=mutated_child(1,109:144);
mutatedchild1_9(1,1:12)=mutated_child(1,145:156);
mutatedchild1_9(1,[places_switch(1,1), places_switch(1,2)])=mutatedchild1_9(1,[places_switch(1,2), places_switch(1,1)]);
for ddd=1:s
    if mutatedchild1_9(1,ddd)==0
        mutatedchild1_10(1,ddd)=0;
    else
        mutatedchild1_10(1,ddd)=15+round(2*rand(1,1));
    end
end
for eee=1:s
    if mutatedchild1_10(1,eee)==0
        mutatedchild1_11(1,eee)=0;
    else
        mutatedchild1_11(1,eee)=transportation(mutatedchild1_8(1,eee+24),mutatedchild1_10(1,eee));
    end
end
mutatedchild1_12(1,1:12)=mutated_child(1,181:192);
for fff=1:s
    if mutatedchild1_9(1,fff)==0
        mutatedchild1_13(1,fff)=transportation(mutatedchild1_8(1,fff+24),mutatedchild1_12(1,fff));
    else
        mutatedchild1_13(1,fff)=transportation(mutatedchild1_10(1,fff),mutatedchild1_12(1,fff));
    end
end
mutatedchild1_14(1,1:12)=mutatedchild1_2(1,1:12);
mutatedchild1_15(1,1:24)=mutated_child(1,217:240);
mutatedchild1_16(1,1:12)=mutatedchild1_6(1,1:12);
for ggg=1:s
    mutatedchild1_17(1,ggg)=transportation(mutatedchild1_15(1,ggg+12),mutatedchild1_16(1,ggg));
end
mutatedchild1_18(1,1:12)=mutatedchild1_7(1,1:12);
mutatedchild1=[mutatedchild1_1 mutatedchild1_2 mutatedchild1_3 mutatedchild1_4 mutatedchild1_5 mutatedchild1_6 mutatedchild1_7 mutatedchild1_8 mutatedchild1_9 mutatedchild1_10 mutatedchild1_11 mutatedchild1_12 mutatedchild1_13 mutatedchild1_14 mutatedchild1_15 mutatedchild1_16 mutatedchild1_17 mutatedchild1_18];
end
children_red(row_mutation(1,1),:)=mutatedchild1(1,:);
suma_residue(1,1)=sum(children_red(popsize,61:72));
best_profit(hhh,1)=profit_sort(1,1);
best_profit2=[best_profit1;best_profit];
for ggh=1:popsize
  for gghh=1:s
    if children_red(ggh,gghh)==3||children_red(ggh,gghh)==4||children_red(ggh,gghh)==5||children_red(ggh,gghh)==6
      children_red(ggh,144+gghh)=0;
    end
  end
end
if hhh==98
  chrom12=children_red;
end

clearvars aaa bbb ccc child child1 child1_1 child1_10 child1_11 child1_12 child1_13 child1_14 child1_15 child1_16 child1_17 child1_18 child1_2 child1_3 child1_4 child1_5 child1_6 child1_7 child1_8 child2 child2_1 child2_10 child2_11 child2_12 child2_13 child2_14 child2_15 child2_16 child2_17 child2_18 child2_2 child2_3 child2_4 child2_5 child2_6 child2_7 child2_8 child2_9 children chrom1 chrom2 chrom3 counter dd ddd ee eee fff gg ggg hh ii in_bio in_bio_sum int_warehouse int_warehouse_sum is2 jj kk
  clearvars ll mm mutated_child mutatedchild1 mutatedchild1_1 mutatedchild1_11 mutatedchild1_12 mutatedchild1_13 mutatedchild1_14 mutatedchild1_15 mutatedchild1_16 mutatedchild1_17 mutatedchild1_18
  clearvars mutatedchild1_2 mutatedchild1_3 mutatedchild1_4 mutatedchild1_5 mutatedchild1_6 mutatedchild1_7 mutatedchild1_8 mutatedchild1_9 mutation_perc nn number_units oo
  clearvars order places places_switch pop2 population pp profit profit_sort qq rank_perc red_pop residue_sum row_mutation rr ss sum_tons1 sum_tons_final1 suma1 switchgrass1
  clearvars total_processed transported1 transported2 transported3 transported4 transported5 transported6 tt uu ww x xx yy zz
  end
plot(best_profit2)
title('Evolution with each Generation');
xlabel('Generation');
ylabel('Profit');
APPENDIX B

Matlab Code for Bee Colony Optimization
clear
clc

months=1:12;
transportation=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
];

alpha=.25;
%0<alpha<=1 due to memory overflow
beta=3;
%0<beta<=20 due to memory overflow
nbees=500;
lambda=.25;
k=.1*1500;
c=size(months,2);
iterations=1500;

for a1=1:nbees
i=1;
for b1=1:c
    pref_path(a1,b1)=randi([i,i+3],1,1);
i=i+4;
end
end
hive=zeros(nbees,1);
pref_path=[hive pref_path];

for bees=1:nbees
    c1=ones(4,1);
    while c1(:,:)<2
        tons=randi([30000,120000],1,12);
        tons=[tons;tons;tons;tons];
        sum_tons=sum(tons,2)*2;
        for r1=1:4
            if sum_tons(r1,1)>2100000
                %              constraint=0
                c1(r1,1)=1;
            else
                %              constraint=1
                c1(r1,1)=2;
            end
        end
    end
end

for a=1:c
    if months(1,a)==3||months(1,a)==4||months(1,a)==5||months(1,a)==6
        btype(1,a)=randi([2,3],1,1);
    else
        btype(1,a)=1;
    end
end
d=randperm(10);
e=1;
for b=1;c
    if btype(1,b)==1
        forigin(1,b)=d(e);
        e=e+1;
    elseif btype(1,b)==2
        forigin(1,b)=randi([11,12],1,1);
    else
        forigin(1,b)=randi([13,14],1,1);
    end
end

tons_resi(:,1:c)=2^.01.*tons(:,1:c);
field_res=randi([1,10],1,12);
intermediate_warehouse=[0 15 16 17];
biorefinery=[20 20 20 20];

for h=1:c
    for g=1:4
        if g==1||h==3||h==4||h==5||h==6
            trans_type1(1,g)=0;
            trans_type2(1,g)=transportation(forigin(1,h),biorefinery(1,g));
        else
            trans_type1(1,g)=transportation(forigin(1,h),intermediate_warehouse(1,g));
            trans_type2(1,g)=transportation(intermediate_warehouse(1,g),biorefinery(1,g));
        end
        trans_typeresi(1,g)=transportation(biorefinery(1,g),field_res(1,h));
    end
    if h==1
        month1=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
    elseif h==2
        month2=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
    elseif h==3
        month3=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
    elseif h==4
        month4=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
    elseif h==5
        month5=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==6
    month6=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==7
    month7=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==8
    month8=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==9
    month9=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==10
    month10=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==11
    month11=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
else
    month12=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
end
end
transportation_month(1:4,1:(4*c))=[month1(:,:), month2(:,:), month3(:,:), month4(:,:), month5(:,:), month6(:,:), month7(:,:), month8(:,:), month9(:,:), month10(:,:), month11(:,:), month12(:,:)];
l=1;

for f=1:c
    for t1=1:4
        if btype(1,f)==1
            switchgrass(t1,f)=tons(t1,f);
            stalk_straw(t1,f)=0;
        else
            switchgrass(t1,f)=0;
            stalk_straw(t1,f)=tons(t1,f);
        end
        number_units(t1,1)=ceil(sum(switchgrass(t1,:)).*2./8./7200);
        cost_hu(t1,1)=580000.*number_units(t1,1);
        hu_month(t1,1)=cost_hu(t1,1)./12;
        harvest_unitcost(t1,f)=hu_month(t1,1).*f;
    end
end
end
ii=4;
for j=1:c
  if j==1
    for k=1:4
      if k==1
        processing_cost(k,j)=50*2*tons(k,j);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j));
        revenue(k,j)=1.8*90*2*tons(k,j);
        if tons(k,j)>60000
          inventory_cost(1,k)=2*(120000+tons_resi(k,j));
        else
          inventory_cost(1,k)=2*(2*tons(k,j)+tons_resi(k,j));
        end
        if transportation_month(3,l)==1
          if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j));
            train_trans(1,k)=0;
          else
            truck_trans(1,k)=0.4*(2*tons(k,j));
            train_trans(1,k)=0.04*tons_resi(k,j);
          end
          else
            if transportation_month(4,l)==2
              truck_trans(1,k)=0;
              train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j));
            else
              truck_trans(1,k)=0.4*tons_resi(k,j);
              train_trans(1,k)=0.04*(2*tons(k,j));
            end
          end
        end
        wb_operation_cost(1,k)=30000*2+1666666.66667;
\[
\text{profit}(1,k) = \text{revenue}(k,j) - \text{processing cost}(k,j) - \text{purchasing cost}(k,j) - \text{inventory cost}(1,k) - \text{truck_trans}(1,k) - \text{train_trans}(1,k) - \text{wb_operation_cost}(1,k) - \text{harvest_unitcost}(k,j);
\]

\[
\text{cost_matrix}(1:9,k) = \left[ \text{revenue}(k,j); \text{processing cost}(k,j); \text{purchasing cost}(k,j); \text{harvest_unitcost}(k,j); \text{inventory cost}(1,k); \text{truck_trans}(1,k); \text{train_trans}(1,k); \text{wb_operation_cost}(1,k); \text{profit}(1,k) \right];
\]

\[
\text{else}
\]
\[
\text{processing cost}(k,j) = 50 \times 2 \times \text{tons}(k,j);
\]
\[
\text{purchasing cost}(k,j) = 2 \times (50 \times \text{switchgrass}(k,j) + 35 \times \text{stalk_straw}(k,j));
\]
\[
\text{revenue}(k,j) = 1.8 \times 90 \times 2 \times \text{tons}(k,j);
\]
\[
\text{if tons}(k,j) > 60000
\]
\[
\text{inventory cost}(1,k) = 2 \times (120000 + \text{tons_resi}(k,j) + \text{tons}(k,j) \times 2);
\]
\[
\text{else}
\]
\[
\text{inventory cost}(1,k) = 2 \times (2 \times \text{tons}(k,j) + \text{tons_resi}(k,j) + \text{tons}(k,j) \times 2);
\]
\[
\text{end}
\]
\[
\text{if transportation_month}(2,l) == 1
\]
\[
\text{if transportation_month}(3,l) == 1
\]
\[
\text{if transportation_month}(4,l) == 1
\]
\[
\text{truck_trans}(1,k) = 0.4 \times (\text{tons_resi}(k,j) + 4 \times \text{tons}(k,j));
\]
\[
\text{train_trans}(1,k) = 0;
\]
\[
\text{else}
\]
\[
\text{truck_trans}(1,k) = 0.4 \times (4 \times \text{tons}(k,j));
\]
\[
\text{train_trans}(1,k) = 0.04 \times \text{tons_resi}(k,j);
\]
\[
\text{end}
\]
\[
\text{else}
\]
\[
\text{if transportation_month}(4,l) == 1
\]
\[
\text{truck_trans}(1,k) = 0.4 \times (\text{tons_resi}(k,j) + 2 \times \text{tons}(k,j));
\]
\[
\text{train_trans}(1,k) = 0.04 \times (2 \times \text{tons}(k,j));
\]
\[
\text{else}
\]
\[
\text{truck_trans}(1,k) = 0.4 \times (2 \times \text{tons}(k,j));
\]
\[
\text{train_trans}(1,k) = 0.04 \times (\text{tons_resi}(k,j) + 2 \times \text{tons}(k,j));
\]
\[
\text{end}
\]
\[
\text{end}
\]
\[
\text{else}
\]
\[
\text{if transportation_month}(3,l) == 1
\]
\[
\text{if transportation_month}(4,l) == 1
\]
\[
110
\]
truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j));
train_trans(1,k)=0.04*(2*tons(k,j));
else
  truck_trans(1,k)=0.4*(2*tons(k,j));
  train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j));
end

train_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j));
else
  if transportation_month(4,l)==1
    truck_trans(1,k)=0.4*(tons_resi(k,j));
    train_trans(1,k)=0.04*(4*tons(k,j));
  else
    truck_trans(1,k)=0;
    train_trans(1,k)=0.04*(tons_resi(k,j)+4*tons(k,j));
  end
end
end

wb_operation_cost(1,k)=30000*2+1666666.66667+60000;
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-
truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
end
l=l+1;
end
hprofit=profit;
r1=1;
p1=2;
for k1=1:4
  pairs(1,[r1,p1])=[pref_path(bees,1) k1];
  [ro1 co1]=find(pairs(1,r1)==pref_path(bees,:));
  [ro2 co2]=find(pairs(1,p1)==pref_path(bees,co1(1,1)+1));
  is1=isempty(co2);
  if is1==0
arcf(1,k1)=lambda;
else
  arcf(1,k1)=(1-lambda)/3;
end
arcf_prof(1,k1)=(arcf(1,k1).^alpha)*(hprofit(1,k1).^beta);
end
for k2=1:4
  probability(1,k2)=arcf_prof(1,k2)/sum(arcf_prof);
end
best_prob=max(probability);
[row,column]=find(probability==best_prob);
if size(column,2)>1
  column_best(1,1)=column(randi([1,size(column,2)],1,1));
else
  column_best=column;
end
if j==3||j==4||j==5||j==6
  column_best=1;
else
  column_best;
end
path(bees,j)=column_best;
tons_path(bees,j)=tons(column_best,j);
best_option_costs(:,j)=cost_matrix(:,column_best);
else
  processing_cost(k,j)=50*2*tons(k,j)+processing_cost(k,j-1);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j))+purchasing_cost(k,j-1);
revenue(k,j)=1.8*90*2*tons(k,j)+revenue(k,j-1);
for k=1:4
  if k==1
    if tons(k,j)>60000
      inventory_cost(1,k)=2*(120000+tons_resi(k,j))+best_option_costs(5,j-1);
    else
      inventory_cost(1,k)=2*(2*tons(k,j)+tons_resi(k,j))+best_option_costs(5,j-1);
    end
  end
end
if transportation_month(3,l)==1
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(6,j-1);
        train_trans(1,k)=0+best_option_costs(7,j-1);
    else
        truck_trans(1,k)=0.4*(2*tons(k,j))+best_option_costs(6,j-1);
        train_trans(1,k)=0.04*tons_resi(k,j)+best_option_costs(7,j-1);
    end
else
    if transportation_month(4,l)==2
        truck_trans(1,k)=0+best_option_costs(6,j-1);
        train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(7,j-1);
    else
        truck_trans(1,k)=0.4*tons_resi(k,j)+best_option_costs(6,j-1);
        train_trans(1,k)=0.04*(2*tons(k,j))+best_option_costs(7,j-1);
    end
end
wb_operation_cost(1,k)=30000*2+833333+best_option_costs(8,j-1);
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);
cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
else
    processing_cost(k,j)=50*2*tons(k,j)+processing_cost(k,j-1);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j))+purchasing_cost(k,j-1);
revenue(k,j)=1.8*90*2*tons(k,j)+revenue(k,j-1);
if tons(k,j)>60000
    inventory_cost(1,k)=2*(120000+tons_resi(k,j)+tons(k,j)*2)+best_option_costs(5,j-1);
else
    inventory_cost(1,k)=2*(2*tons(k,j)+tons_resi(k,j)+tons(k,j)*2)+best_option_costs(5,j-1);
end
if transportation_month(2,l)==1
    if transportation_month(3,l)==1
        if transportation_month(4,l)==1
truck_trans(1,k)=0.4*(tons_resi(k,j)+4*tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0+best_option_costs(7,j-1);
else
truck_trans(1,k)=0.4*(4*tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*tons_resi(k,j)+best_option_costs(7,j-1);
end
else
if transportation_month(4,l)==1
truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(2*tons(k,j))+best_option_costs(7,j-1);
else
truck_trans(1,k)=0.4*(2*tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(7,j-1);
end
end
else
if transportation_month(3,l)==1
if transportation_month(4,l)==1
truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(2*tons(k,j))+best_option_costs(7,j-1);
else
truck_trans(1,k)=0.4*(2*tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(7,j-1);
end
else
if transportation_month(4,l)==1
truck_trans(1,k)=0.4*(tons_resi(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(4*tons(k,j))+best_option_costs(7,j-1);
else
truck_trans(1,k)=0+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(tons_resi(k,j)+4*tons(k,j))+best_option_costs(7,j-1);
end
end
end
\[
\text{wb\_operation\_cost}(1,k) = 30000 \times 2 + 166666.6667 + 60000 + \text{best\_option\_costs}(8,j-1);
\]
\[
\text{profit}(1,k) = \text{revenue}(k,j) - \text{processing\_cost}(k,j) - \text{purchasing\_cost}(k,j) - \text{inventory\_cost}(1,k) - \text{truck\_trans}(1,k) - \text{train\_trans}(1,k) - \text{wb\_operation\_cost}(1,k) - \text{harvest\_unitcost}(k,j);
\]
\[
\text{cost\_matrix}(1:9,k) = [\text{revenue}(k,j) ; \text{processing\_cost}(k,j) ; \text{purchasing\_cost}(k,j) ; \text{harvest\_unitcost}(k,j) ; \text{inventory\_cost}(1,k) ; \text{truck\_trans}(1,k) ; \text{train\_trans}(1,k) ; \text{wb\_operation\_cost}(1,k) ; \text{profit}(1,k)];
\]
\[
\text{end}
\]
\[
l = l + 1;
\]
\[
\text{end}
\]
\[
\text{hprofit} = \text{profit};
\]
\[
r1 = 1;
\]
\[
p1 = 2;
\]
\[
\text{for } k1 = 1:4
\]
\[
\text{pairs}(1,[r1,p1]) = [\text{path}(bees,j-1) k1+ii];
\]
\[
[ro1,co1] = \text{find}(\text{pairs}(1,r1) == \text{pref\_path}(bees,:));
\]
\[
is2 = \text{isempty}(co1);
\]
\[
\text{if } is2 == 0
\]
\[
[ro2,co2] = \text{find}(\text{pairs}(1,p1) == \text{pref\_path}(bees,co1(1,1)+1));
\]
\[
is1 = \text{isempty}(co2);
\]
\[
\text{if } is1 == 0
\]
\[
\text{arcf}(1,k1) = \lambda;
\]
\[
\text{else}
\]
\[
\text{arcf}(1,k1) = (1-\lambda)/3;
\]
\[
\text{end}
\]
\[
\text{else}
\]
\[
\text{arcf}(1,k1) = 1/4;
\]
\[
\text{end}
\]
\[
\text{arcf\_prof}(1,k1) = (\text{arcf}(1,k1)^\alpha) * (\text{hprofit}(1,k1)^\beta);
\]
\[
\text{end}
\]
\[
\text{for } k2 = 1:4
\]
\[
\text{probability}(1,k2) = \text{arcf\_prof}(1,k2)/\text{sum(\text{arcf\_prof})};
\]
\[
\text{end}
\]
\[
\text{best\_prob} = \text{max}(\text{probability});
\]
\[
[\text{row, column}] = \text{find}(\text{probability} == \text{best\_prob});
\]
if size(column,2)>1
    column_best(1,1)=column(randi([1,size(column,2)],1,1));
else
    column_best=column;
end
if j==3||j==4||j==5||j==6
    column_best=1;
else
    column_best=column_best;
end
path(bees,j)=column_best+ii;
ii=ii+4;
tons_path(bees,j)=tons(column_best,j);
best_option_costs(:,j)=cost_matrix(:,column_best);
end
end
profit_bees(bees,1)=best_option_costs(9,12);
end
clearvars -except c k profit_bees tons_path path nbees lambda c months transportation alpha beta hive pref_path iterations

%%Profit Pref_Path
for bees=1:nbees
    c1=1;
    while c1(:,:)<2
        pref_path_tons(1,1:12)=randi([30000,120000],1,12);
        sum_tons=sum(pref_path_tons,2);
        if sum_tons>2100000
            % constraint=0
            c1=1;
        else
            % constraint=1
            c1=2;
        end
    end
end
for a=1:c
    if months(1,a)==3||months(1,a)==4||months(1,a)==5||months(1,a)==6
        btype(1,a)=randi([2,3],1,1);
    else
        btype(1,a)=1;
    end
end
d=randperm(10);
e=1;
for b=1:c
    if btype(1,b)==1
        forigin(1,b)=d(e);
e=e+1;
    elseif btype(1,b)==2
        forigin(1,b)=randi([11,12],1,1);
    else
        forigin(1,b)=randi([13,14],1,1);
    end
end
tons_resi(:,1:c)=2*.01.*pref_path_tons(:,1:c);
field_res=randi([1,10],1,12);

% intermediate_warehouse=[0 15 16 17]
% biorefinery=[20 20 20 20]
pref_path1=pref_path(:,(2:13));
pp1=1:4:45;
pp2=2:4:46;
pp3=3:4:47;
pp4=4:4:48;
for h=1:c
    if any(pref_path1(bees,h)==pp1)==1
        intermediate_warehouse=0;
        trans_type1=0;
        trans_type2=transportation(forigin(1,h),20);
    elseif any(pref_path1(bees,h)==pp2)==1
        intermediate_warehouse=15;
trans_type1=transportation(forigin(1,h),intermediate_warehouse);
trans_type2=transportation(intermediate_warehouse,20);
elseif any(pref_path1(bees,h)==pp3)==1
    intermediate_warehouse=16;
    trans_type1=transportation(forigin(1,h),intermediate_warehouse);
    trans_type2=transportation(intermediate_warehouse,20);
else
    intermediate_warehouse=17;
    trans_type1=transportation(forigin(1,h),intermediate_warehouse);
    trans_type2=transportation(intermediate_warehouse,20);
end

trans_typeresi=transportation(20,field_res(1,h));

if h==1
    month1=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==2
    month2=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==3
    month3=[0; trans_type1; trans_type2; trans_typeresi];
elseif h==4
    month4=[0; trans_type1; trans_type2; trans_typeresi];
elseif h==5
    month5=[0; trans_type1; trans_type2; trans_typeresi];
elseif h==6
    month6=[0; trans_type1; trans_type2; trans_typeresi];
elseif h==7
    month7=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==8
    month8=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==9
    month9=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==10
    month10=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==11
    month11=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
elseif h==12
    month12=[intermediate_warehouse; trans_type1; trans_type2; trans_typeresi];
end
month11=[intermediate_warehouse; trans_type1; trans_type2; trans_type3]
else
month12=[intermediate_warehouse; trans_type1; trans_type2; trans_type3]
end
end
transportation_month(1:4,1:12)=[month1(1,:), month2(1,:), month3(1,:), month4(1,:), month5(1,:),
month6(1,:), month7(1,:), month8(1,:), month9(1,:), month10(1,:), month11(1,:), month12(1,:)]
l=1;
%
for f=1:c
if btype(1,f)==1
switchgrass(1,f)=pref_path_tons(1,f);
stalk_straw(1,f)=0;
else
switchgrass(1,f)=0;
stalk_straw(1,f)=pref_path_tons(1,f);
end
number_units(1,1)=ceil(sum(switchgrass(1,:)).*2./8./7200);
cost_hu(1,1)=580000.*number_units(1,1);
hu_month(1,1)=cost_hu(1,1)./12;
harvest_unitcost(1,f)=hu_month(1,1).*f;
end
ii=4;
k=1;
for j=1:c
if j==1
if transportation_month(1,j)==0
processing_cost(k,j)=50*2*pref_path_tons(k,j);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j));
revenue(k,j)=1.8*90*2*pref_path_tons(k,j);
if pref_path_tons(k,j)>60000
inventory_cost(1,k)=2*(120000+tons_resi(k,j));
else
inventory_cost(1,k)=2*(2*pref_path_tons(k,j)+tons_resi(k,j));
end
if transportation_month(3,l)==1
if transportation_month(4,l)==1
  truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j));
  train_trans(1,k)=0;
else
  truck_trans(1,k)=0.4*(2*pref_path_tons(k,j));
  train_trans(1,k)=0.04*tons_resi(k,j);
end
else
  if transportation_month(4,l)==2
    truck_trans(1,k)=0;
    train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j));
  else
    truck_trans(1,k)=0.4*tons_resi(k,j);
    train_trans(1,k)=0.04*(2*pref_path_tons(k,j));
  end
end
wb_operation_cost(1,k)=30000*2+1666666.66667;
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
else
  processing_cost(k,j)=50*2*pref_path_tons(k,j);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j));
revenue(k,j)=1.8*90*2*pref_path_tons(k,j);
if pref_path_tons(k,j)>60000
  inventory_cost(1,k)=2*(120000+tons_resi(k,j)+pref_path_tons(k,j)*2);
else
  inventory_cost(1,k)=2*(2*pref_path_tons(k,j)+tons_resi(k,j)+pref_path_tons(k,j)*2);
end
if transportation_month(2,l)==1
if transportation_month(3,l)==1
  if transportation_month(4,l)==1
    truck_trans(1,k)=0.4*(tons_resi(k,j)+4*pref_path_tons(k,j));
    train_trans(1,k)=0;
  else
    truck_trans(1,k)=0.4*(4*pref_path_tons(k,j));
    train_trans(1,k)=0.04*tons_resi(k,j);
  end
else
  if transportation_month(4,l)==1
    truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j));
    train_trans(1,k)=0.04*(2*pref_path_tons(k,j));
  else
    truck_trans(1,k)=0.4*(2*pref_path_tons(k,j));
    train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j));
  end
end
else
  if transportation_month(3,l)==1
    if transportation_month(4,l)==1
      truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j));
      train_trans(1,k)=0.04*(2*pref_path_tons(k,j));
    else
      truck_trans(1,k)=0.4*(2*pref_path_tons(k,j));
      train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j));
    end
    train_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j));
  else
    if transportation_month(4,l)==1
      truck_trans(1,k)=0.4*(tons_resi(k,j));
      train_trans(1,k)=0.04*(4*pref_path_tons(k,j));
    else
      truck_trans(1,k)=0;
      train_trans(1,k)=0.04*(tons_resi(k,j)+4*pref_path_tons(k,j));
    end
  end
end
end

wb_operation_cost(1,k)=30000*2+1666666.66667+60000;
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-
truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);
truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];

end

l=l+1;

best_option_costs(1:9,j)=cost_matrix;
else

processing_cost(k,j)=50*2*pref_path_tons(k,j)+processing_cost(k,j-1);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j))+purchasing_cost(k,j-1);
revenue(k,j)=1.8*90*2*pref_path_tons(k,j)+revenue(k,j-1);

if transportation_month(1,j)==0
    if pref_path_tons(k,j)>60000
        inventory_cost(1,k)=2*(120000+tons_resi(k,j))+best_option_costs(5,j-1);
    else
        inventory_cost(1,k)=2*(2*pref_path_tons(k,j)+tons_resi(k,j))+best_option_costs(5,j-1);
    end

if transportation_month(3,l)==1
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs(6,j-1);
        train_trans(1,k)=0+best_option_costs(7,j-1);
    else
        truck_trans(1,k)=0.4*(2*pref_path_tons(k,j))+best_option_costs(6,j-1);
        train_trans(1,k)=0.04*tons_resi(k,j)+best_option_costs(7,j-1);
    end

else
    if transportation_month(4,l)==2
        truck_trans(1,k)=0+best_option_costs(6,j-1);
        train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs(7,j-1);
    else

122
\[
\text{truck}_\text{trans}(1,k) = 0.4*\text{tons}_\text{resi}(k,j) + \text{best}_\text{option}_\text{costs}(6,j-1); \\
\text{train}_\text{trans}(1,k) = 0.04*(2*\text{pref}_\text{path}_\text{tons}(k,j)) + \text{best}_\text{option}_\text{costs}(7,j-1); \\
\text{end}
\]
end

\[
\text{wb}_\text{operation}_\text{cost}(1,k) = 30000*2 + 1666666.66667 + \text{best}_\text{option}_\text{costs}(8,j-1); \\
\text{profit}(1,k) = \text{revenue}(k,j) - \text{processing}_\text{cost}(k,j) - \text{purchasing}_\text{cost}(k,j) - \text{inventory}_\text{cost}(1,k) - \text{truck}_\text{trans}(1,k) - \text{train}_\text{trans}(1,k) - \text{wb}_\text{operation}_\text{cost}(1,k) - \text{harvest}_\text{unitcost}(k,j); \\
\text{cost}_\text{matrix}(1:9,k) = [\text{revenue}(k,j); \text{processing}_\text{cost}(k,j); \text{purchasing}_\text{cost}(k,j); \text{harvest}_\text{unitcost}(k,j); \text{inventory}_\text{cost}(1,k); \text{truck}_\text{trans}(1,k); \text{train}_\text{trans}(1,k); \text{wb}_\text{operation}_\text{cost}(1,k); \text{profit}(1,k)]; \\
\text{else}
\]
\[
\text{processing}_\text{cost}(k,j) = 50*2*\text{pref}_\text{path}_\text{tons}(k,j) + \text{processing}_\text{cost}(k,j-1); \\
\text{purchasing}_\text{cost}(k,j) = 2*(50*\text{switchgrass}(k,j)+35*\text{stalk}_\text{straw}(k,j)) + \text{purchasing}_\text{cost}(k,j-1); \\
\text{revenue}(k,j) = 1.8*90*2*\text{pref}_\text{path}_\text{tons}(k,j) + \text{revenue}(k,j-1); \\
\text{if} \quad \text{pref}_\text{path}_\text{tons}(k,j) > 60000
\]
\[
\text{inventory}_\text{cost}(1,k) = 2*(120000+\text{tons}_\text{resi}(k,j)+\text{pref}_\text{path}_\text{tons}(k,j)*2)+\text{best}_\text{option}_\text{costs}(5,j-1); \\
\text{else}
\]
\[
\text{inventory}_\text{cost}(1,k) = 2*(2*\text{pref}_\text{path}_\text{tons}(k,j)+\text{tons}_\text{resi}(k,j)+\text{pref}_\text{path}_\text{tons}(k,j)*2)+\text{best}_\text{option}_\text{costs}(5,j-1); \\
\text{end}
\]
\text{if} \quad \text{transportation}_\text{month}(2,l) == 1
\]
\text{if} \quad \text{transportation}_\text{month}(3,l) == 1
\]
\text{if} \quad \text{transportation}_\text{month}(4,l) == 1
\]
\[
\text{truck}_\text{trans}(1,k) = 0.4*(\text{tons}_\text{resi}(k,j) + 4*\text{pref}_\text{path}_\text{tons}(k,j)) + \text{best}_\text{option}_\text{costs}(6,j-1); \\
\text{train}_\text{trans}(1,k) = 0 + \text{best}_\text{option}_\text{costs}(7,j-1); \\
\text{else}
\]
\[
\text{truck}_\text{trans}(1,k) = 0.4*(4*\text{pref}_\text{path}_\text{tons}(k,j)) + \text{best}_\text{option}_\text{costs}(6,j-1); \\
\text{train}_\text{trans}(1,k) = 0.04*\text{tons}_\text{resi}(k,j) + \text{best}_\text{option}_\text{costs}(7,j-1); \\
\text{end}
\]
\text{else}
\]
\text{if} \quad \text{transportation}_\text{month}(4,l) == 1
\]
\[
\text{truck}_\text{trans}(1,k) = 0.4*(\text{tons}_\text{resi}(k,j) + 2*\text{pref}_\text{path}_\text{tons}(k,j)) + \text{best}_\text{option}_\text{costs}(6,j-1); \\
\text{train}_\text{trans}(1,k) = 0.04*(2*\text{pref}_\text{path}_\text{tons}(k,j)) + \text{best}_\text{option}_\text{costs}(7,j-1); \\
\text{else}
truck_trans(1,k)=0.4*(2*pref_path_tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs(7,j-1);
end
end
else
if transportation_month(3,l)==1
if transportation_month(4,l)==1
truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(2*pref_path_tons(k,j))+best_option_costs(7,j-1);
else
truck_trans(1,k)=0.4*(2*pref_path_tons(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs(7,j-1);
end
else
if transportation_month(4,l)==1
truck_trans(1,k)=0.4*(tons_resi(k,j))+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(4*pref_path_tons(k,j))+best_option_costs(7,j-1);
else
truck_trans(1,k)=0+best_option_costs(6,j-1);
train_trans(1,k)=0.04*(tons_resi(k,j)+4*pref_path_tons(k,j))+best_option_costs(7,j-1);
end
end
end
wb_operation_cost(1,k)=30000*2+1666666.66667+60000+best_option_costs(8,j-1);
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);
cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
end
l=l+1;
best_option_costs(1:9,j)=cost_matrix;
end
end
pref_path_tons1(bees,1:12)=pref_path_tons;
hprofit1(bees,1)=best_option_costs(9,12);
end
clearvars -except best_profit_bees1 hprofit1 c k hprofit pref_path1 pref_path_tons1 profit_bees tons_path path nbees lambda c months transportation alpha beta hive pref_path iterations

% for bees=1:nbees
%     if profit_bees(bees,1)<hprofit(bees,1)
%         final_profit(bees,1)=hprofit(bees,1)
%         final_path(bees,1:12)=pref_path1(bees,1:12)
%         final_tons(bees,1:12)=pref_path_tons1(bees,1:12)
%     else
%         final_profit(bees,1)=profit_bees(bees,1)
%         final_path(bees,1:12)=path(bees,1:12)
%         final_tons(bees,1:12)=tons_path(bees,1:12)
%     end
% end
for bees=1:nbees
    if profit_bees(bees,1)>hprofit1(bees,1)
        nwaggle(bees,1)=bees;
    else
        nwaggle(bees,1)=0;
    end
end
nwaggle1=nwaggle;
nwaggle1(nwaggle1==0)=[];
is3=isempty(nwaggle1);
if is3==1
    pfollow=zeros(size(nwaggle,1),1);
else
    for zx=1:size(nwaggle1,1)
nwaggledur(zx,1:25)=[path(nwaggle1(zx,1,:));
tons_path(nwaggle1(zx,1,:));
profit_bees(nwaggle1(zx,1,:));
end
nwaggledur1=[nwaggledur zeros(size(nwaggledur,1),1)];
for r=1:size(nwaggle,1)
  if nwaggle(r,:)==0
    pfbee(r,1)=0;
  else
    pfbee(r,1)=profit_bees(nwaggle(r,:,:));
  end
end
pfcolony=(1/size(nwaggle1,1))*sum(pfbee);
for tpr=1:size(nwaggle,1)
  if nwaggle(tpr,1)==0
    duration(tpr,1)=0;
  else
    duration(tpr,1)=k*(pfbee(tpr,:)/pfcolony);
  end
end
duration1=duration;
duration1(duration1==0)=[ ];
duration1=ceil(duration1);
nwaggledur1(:,26)=[duration1];
for rs=1:size(nwaggle,1)
  if nwaggle(rs,1)==0
    pfollow(rs,:)=0;
  elseif pfbee(rs,:)<.95*pfcolony
    pfollow(rs,:)=.6;
  elseif .95*pfcolony<pfbee(rs,:)<.975*pfcolony
    pfollow(rs,:)=.2;
  elseif .975*pfcolony<pfbee(rs,:)<.99*pfcolony
    pfollow(rs,:)=.02;
  else
    pfollow(rs,:)=0;
  end
end
end
def
pref_path_tons2=pref_path_tons1;
best_profit_bees1(1,1)=max(profit_bees);
clearvars -except nwaggledur1 best_profit_bees1 hprofit1 c pfollow pfbee duartion1 nwaggle nwaggle1 k hprofit pref_path1 pref_path_tons1 profit_bees tons_path path nbees lambda c months transportation alpha beta hive iterations pref_path_tons2
%2nd Iteration
tons_final=[];
tons_resi_all=[];
paths_profit_all=[];
best_option_cost_all=[];
forigin_all=[];
transportation_month_all=[];
for v=1:iterations-1

size1=size(pfollow(:,,:),1);
missing=nbees-size1;
if size1>=nbees
    pfollow=pfollow(1:nbees,:);
else
    pfollow=[pfollow(:,,:);zeros(missing,1)]
end

for bees=1:nbees
    i=1;
    if pfollow(bees,1)==0
        for b1=1:c
            pref_path(bees,b1)=randi([i,i+3],1,1);
            i=i+4;
        end
        hprofit1(bees,1)=0;
    else
        random=rand(1);
end
if random(1,1)<pfollow(bees,1)
    pref_path(bees,1:12)=pref_path1(bees,1:12);
    hprofit1(bees,1)=hprofit1(bees,1);
else
    for b1=1:c
        pref_path(bees,b1)=randi([i,i+3],1,1);
        i=i+4;
    end
    hprofit1(bees,1)=0;
end
eend
end
hive=zeros(nbees,1);
pref_path=[hive pref_path];
for bees=1:nbees
    c1=ones(4,1);
    while c1(:,1)<2
        tons=randi([30000,120000],1,12);
        tons=[tons;tons;tons;tons];
        sum_tons=sum(tons,2)*2;
        for r1=1:4
            if sum_tons(r1,1)>2100000
                %            constraint=0
                c1(r1,1)=1;
            else
                %            constraint=1
                c1(r1,1)=2;
            end
        end
    end
for a=1:c
    if months(1,a)==3||months(1,a)==4||months(1,a)==5||months(1,a)==6
        btype(1,a)=randi([2,3],1,1);
    else
        %            constraint=0
        btype(1,a)=randi([2,3],1,1);
    end
end
btype(1,a)=1;
end

d=randperm(10);
e=1;
for b=1:c
    if btype(1,b)==1
        forigin(1,b)=d(e);
e=e+1;
    elseif btype(1,b)==2
        forigin(1,b)=randi([11,12],1,1);
    else
        forigin(1,b)=randi([13,14],1,1);
    end
end

tons_resi(:,1:c)=2*.01.*tons(:,1:c);
field_res=randi([1,10],1,12);

intermediate_warehouse=[0 15 16 17];
biorefinery=[20 20 20 20];

for h=1:c
    for g=1:4
        if g==1||h==3||h==5||h==6
            trans_type1(1,g)=0;
            trans_type2(1,g)=transportation(forigin(1,h),biorefinery(1,g));
        else
            trans_type1(1,g)=transportation(forigin(1,h),intermediate_warehouse(1,g));
            trans_type2(1,g)=transportation(intermediate_warehouse(1,g),biorefinery(1,g));
        end
        trans_type3er(1,g)=transportation(biorefinery(1,g),field_res(1,h));
    end
end

if h==1
    month1=[intermediate_warehouse; trans_type1; trans_type2; trans_type3er];
elseif h==2
    month2=[intermediate_warehouse; trans_type1; trans_type2; trans_type3er];
elseif h==3

month3=[zeros(1,4); trans_type1; trans_type2; trans_type3];
elseif h==4
    month4=[zeros(1,4); trans_type1; trans_type2; trans_type3];
elseif h==5
    month5=[zeros(1,4); trans_type1; trans_type2; trans_type3];
elseif h==6
    month6=[zeros(1,4); trans_type1; trans_type2; trans_type3];
elseif h==7
    month7=[intermediate_warehouse; trans_type1; trans_type2; trans_type3];
elseif h==8
    month8=[intermediate_warehouse; trans_type1; trans_type2; trans_type3];
elseif h==9
    month9=[intermediate_warehouse; trans_type1; trans_type2; trans_type3];
elseif h==10
    month10=[intermediate_warehouse; trans_type1; trans_type2; trans_type3];
elseif h==11
    month11=[intermediate_warehouse; trans_type1; trans_type2; trans_type3];
else
    month12=[intermediate_warehouse; trans_type1; trans_type2; trans_type3];
end

transportation_month(1:4,1:(4*c))=[month1(:,1), month2(:,1), month3(:,1), month4(:,1), month5(:,1), month6(:,1), month7(:,1), month8(:,1), month9(:,1), month10(:,1), month11(:,1), month12(:,1)];
l=1;

for f=1:c
    for t1=1:4
        if btype(1,f)==1
            switchgrass(t1,f)=tons(t1,f);
            stalk_straw(t1,f)=0;
        else
            switchgrass(t1,f)=0;
            stalk_straw(t1,f)=tons(t1,f);
        end
    number_units(t1,1)=ceil(sum(switchgrass(t1,:)).*2./8./7200);
cost_hu(t1,1)=580000.*number_units(t1,1);
hu_month(t1,1)=cost_hu(t1,1)./12;
harvest_unitcost(t1,f)=hu_month(t1,1).*f;
end
end
ii=4;
for j=1:c
    if j==1
        for k=1:4
            if k==1
                processing_cost(k,j)=50*2*tons(k,j);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j));
revenue(k,j)=1.8*90*2*tons(k,j);

                if tons(k,j)>60000
                    inventory_cost(1,k)=2*(120000+tons_resi(k,j));
                else
                    inventory_cost(1,k)=2*(2*tons(k,j)+tons_resi(k,j));
                end

                if transportation_month(3,l)==1
                    if transportation_month(4,l)==1
                        truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j));
                        train_trans(1,k)=0;
                    else
                        truck_trans(1,k)=0.4*(2*tons(k,j));
                        train_trans(1,k)=0.04*tons_resi(k,j);
                    end
                else
                    if transportation_month(4,l)==2
                        truck_trans(1,k)=0;
                        train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j));
                    else
                        truck_trans(1,k)=0.4*tons_resi(k,j);
                    end
                end
            else
            end
        end
    end
end
train_trans(1,k)=0.04*(2*tons(k,j));
end
end
wb_operation_cost(1,k)=30000*2+1666666.66667;
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);
cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
else
    processing_cost(k,j)=50*2*tons(k,j);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j));
revenue(k,j)=1.8*90*2*tons(k,j);
if tons(k,j)>60000
    inventory_cost(1,k)=2*(120000+tons_resi(k,j)+tons(k,j)*2);
else
    inventory_cost(1,k)=2*(2*tons(k,j)+tons_resi(k,j)+tons(k,j)*2);
end
if transportation_month(2,l)==1
    if transportation_month(3,l)==1
        if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j)+4*tons(k,j));
            train_trans(1,k)=0;
        else
            truck_trans(1,k)=0.4*(4*tons(k,j));
            train_trans(1,k)=0.04*tons_resi(k,j);
        end
    else
        truck_trans(1,k)=0.4*(4*tons(k,j));
        train_trans(1,k)=0.04*tons_resi(k,j);
    end
else
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j));
        train_trans(1,k)=0.04*(2*tons(k,j));
    else
        truck_trans(1,k)=0.4*(2*tons(k,j));
        train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j));
    end
end
if transportation_month(3,l)==1
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j));
        train_trans(1,k)=0.04*(2*tons(k,j));
    else
        truck_trans(1,k)=0.4*(2*tons(k,j));
        train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j));
    end
    train_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j));
else
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j));
        train_trans(1,k)=0.04*(4*tons(k,j));
    else
        truck_trans(1,k)=0;
        train_trans(1,k)=0.04*(tons_resi(k,j)+4*tons(k,j));
    end
end

wb_operation_cost(1,k)=30000*2+1666666.66667+60000;
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-
trick_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];

end
l=l+1;
end

hprofit=profit;
r1=1;
p1=2;
for k1=1:4
    pairs(1,[r1,p1])=[pref_path(bees,1) k1];
end
\[ \text{ro1 co1} = \text{find(pairs(1,r1)==pref_path(bees,:));} \]
\[ \text{ro2 co2} = \text{find(pairs(1,p1)==pref_path(bees,co1(1,1)+1));} \]
\[ \text{is1} = \text{isempty(co2);} \]
\[ \text{if is1==0} \]
\[ \text{arcf}(1,k1) = \lambda; \]
\[ \text{else} \]
\[ \text{arcf}(1,k1) = (1-\lambda)/3; \]
\[ \text{end} \]
\[ \text{arcf}_\text{prof}(1,k1) = (\text{arcf}(1,k1)^{\alpha})(\text{hprofit}(1,k1)^{\beta}); \]
\[ \text{end} \]
\[ \text{for k2=1:4} \]
\[ \text{probability}(1,k2) = \text{arcf}_\text{prof}(1,k2)/\text{sum(arcf}_\text{prof);} \]
\[ \text{end} \]
\[ \text{best_prob} = \text{max(probability);} \]
\[ \text{[row,column] = find(probability==best_prob);} \]
\[ \text{if size(column,2)>1} \]
\[ \text{column_best}(1,1) = \text{column(randi([1,size(column,2)],1,1));} \]
\[ \text{else} \]
\[ \text{column_best} = \text{column;} \]
\[ \text{end} \]
\[ \text{if j==3||j==4||j==5||j==6} \]
\[ \text{column_best} = 1; \]
\[ \text{else} \]
\[ \text{column_best} = \text{column_best;} \]
\[ \text{end} \]
\[ \text{path(bees,j) = column_best;} \]
\[ \text{tons}_\text{path}(bees,j) = \text{tons(column_best,j);} \]
\[ \text{best_option_costs(:,j) = cost_matrix(:,column_best);} \]
\[ \text{else} \]
\[ \text{for k=1:4} \]
\[ \text{if k==1} \]
\[ \text{processing_cost(k,j) = 50*2*tons(k,j)+best_option_costs(2,j-1);} \]
\[ \text{purchasing_cost(k,j) = 2*(50*switchgrass(k,j)+35*stalk_straw(k,j))+best_option_costs(3,j-1);} \]
revenue(k,j) = 1.8*90*2*tons(k,j) + best_option_costs(1,j-1);
if tons(k,j) > 60000

inventory_cost(1,k) = 2*(120000 + tons_resi(k,j)) + best_option_costs(5,j-1);
else
inventory_cost(1,k) = 2*(2*tons(k,j) + tons_resi(k,j)) + best_option_costs(5,j-1);
end
if transportation_month(3,l) == 1
if transportation_month(4,l) == 1
truck_trans(1,k) = 0.4*(tons_resi(k,j) + 2*tons(k,j)) + best_option_costs(6,j-1);
train_trans(1,k) = 0 + best_option_costs(7,j-1);
else
truck_trans(1,k) = 0.4*(2*tons(k,j)) + best_option_costs(6,j-1);
train_trans(1,k) = 0.04*tons_resi(k,j) + best_option_costs(7,j-1);
end
else
if transportation_month(4,l) == 2
truck_trans(1,k) = 0 + best_option_costs(6,j-1);
train_trans(1,k) = 0.04*(tons_resi(k,j) + 2*tons(k,j)) + best_option_costs(7,j-1);
else
truck_trans(1,k) = 0.4*tons_resi(k,j) + best_option_costs(6,j-1);
train_trans(1,k) = 0.04*(2*tons(k,j)) + best_option_costs(7,j-1);
end
end
wb_operation_cost(1,k) = 30000*2 + 1666666.66667 + best_option_costs(8,j-1);
profit(1,k) = revenue(k,j) - processing_cost(k,j) - purchasing_cost(k,j) - inventory_cost(1,k) - truck_trans(1,k) - train_trans(1,k) - wb_operation_cost(1,k) - harvest_unitcost(k,j);

else
processing_cost(k,j) = 50*2*tons(k,j) + best_option_costs(2,j-1);
purchasing_cost(k,j) = 2*(50*switchgrass(k,j) + 35*stalk_straw(k,j)) + best_option_costs(3,j-1);
revenue(k,j) = 1.8*90*2*tons(k,j) + best_option_costs(1,j-1);
if tons(k,j)>60000
    inventory_cost(1,k)=2*(120000+tons_resi(k,j)+tons(k,j)*2)+best_option_costs(5,j-1);
else
    inventory_cost(1,k)=2*(2*tons(k,j)+tons_resi(k,j)+tons(k,j)*2)+best_option_costs(5,j-1);
end

if transportation_month(2,l)==1
    if transportation_month(3,l)==1
        if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j)+4*tons(k,j))+best_option_costs(6,j-1);
            train_trans(1,k)=0+best_option_costs(7,j-1);
        else
            truck_trans(1,k)=0.4*(4*tons(k,j))+best_option_costs(6,j-1);
            train_trans(1,k)=0.04*tons_resi(k,j)+best_option_costs(7,j-1);
        end
    else
        if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(6,j-1);
            train_trans(1,k)=0.04*(2*tons(k,j))+best_option_costs(7,j-1);
        else
            truck_trans(1,k)=0.4*(2*tons(k,j))+best_option_costs(6,j-1);
            train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(7,j-1);
        end
    end
else
    if transportation_month(3,l)==1
        if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(6,j-1);
            train_trans(1,k)=0.04*(2*tons(k,j))+best_option_costs(7,j-1);
        else
            truck_trans(1,k)=0.4*(2*tons(k,j))+best_option_costs(6,j-1);
            train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(7,j-1);
        end
    end
else
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(6,j-1);
        train_trans(1,k)=0.04*(2*tons(k,j))+best_option_costs(7,j-1);
    else
        truck_trans(1,k)=0.4*(2*tons(k,j))+best_option_costs(6,j-1);
        train_trans(1,k)=0.04*(tons_resi(k,j)+2*tons(k,j))+best_option_costs(7,j-1);
    end
end
else
if transportation_month(4,l)==1
  truck_trans(1,k)=0.4*(tons_resi(k,j))+best_option_costs(6,j-1);
  train_trans(1,k)=0.04*(4*tons(k,j))+best_option_costs(7,j-1);
else
  truck_trans(1,k)=0+best_option_costs(6,j-1);
  train_trans(1,k)=0.04*(tons_resi(k,j)+4*tons(k,j))+best_option_costs(7,j-1);
end
end
d
wb_operation_cost(1,k)=30000*2+1666666.66667+60000+best_option_costs(8,j-1);
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-tr
 uck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);
cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitc
  ost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);
end
l=l+1;
end
hprofit=profit;
r1=1;
p1=2;
for k1=1:4
  pairs(1,[r1,p1])=[path(bees,j-1) k1+ii];
  [ro1 co1]=find(pairs(1,r1)==pref_path(bees,:));
  is2=isempty(co1);
  if is2==0
    [ro2 co2]=find(pairs(1,p1)==pref_path(bees,co1(1,1)+1));
    is1=isempty(co2);
    if is1==0
      arcf(1,k1)=lambda;
    else
      arcf(1,k1)=(1-lambda)/3;
    end
  else
    arcf(1,k1)=1/4;
arcf_prof(1,k1) = (arcf(1,k1)^alpha) * (hprofit(1,k1)^beta);

for k2 = 1:4
    probability(1,k2) = arcf_prof(1,k2) / sum(arcf_prof);
end

best_prob = max(probability);
[row, column] = find(probability == best_prob);
if size(column,2) > 1
    column_best(1,1) = column(randi([1, size(column,2)], 1, 1));
else
    column_best = column;
end

if j == 3 || j == 4 || j == 5 || j == 6
    column_best = 1;
else
    column_best = column_best;
end

path(bees,j) = column_best + ii;
ii = ii + 4;

tons_path(bees,j) = tons(column_best, j);
best_option_costs(:, j) = cost_matrix(:, column_best);
end

profit_bees(bees,1) = best_option_costs(9,12);

paths_profit = [path profit_bees];
tons_final = [tons_final; tons_path];
paths_profit_all = [paths_profit_all; paths_profit];
best_option_cost_all = [best_option_cost_all; best_option_costs];
forigin_all = [forigin_all; forigin];
transportation_month_all = [transportation_month_all; transportation_month];
clearvars -except tons_final tons_resi_all paths_profit_all best_option_cost_all forigin_all transportation_month_all nwaggledur1 v best_profit_bees best_profit_bees1 hprofit1 pref_path_tons2 c k profit_bees tons_path path nbees lambda c months transportation alpha beta hive pref_path iterations hprofit

for bees=1:nbees
  c1=1;
  if hprofit1(bees,1)==0
    while c1(:,:)<2
      pref_path_tons(1,1:12)=randi([30000,120000],1,12);
      sum_tons=sum(pref_path_tons,2);
      if sum_tons>2100000
        constraint=0;
        c1=1;
      else
        constraint=1;
        c1=2;
      end
    end
  end
  for a=1:c
    if months(1,a)==3||months(1,a)==4||months(1,a)==5||months(1,a)==6
      btype(1,a)=randi([2,3],1,1);
    else
      btype(1,a)=1;
    end
  end
  d=randperm(10);
  e=1;
  for b=1:c
    if btype(1,b)==1
      forigin(1,b)=d(e);
      e=e+1;
    elseif btype(1,b)==2
      forigin(1,b)=randi([11,12],1,1);
    else
      forgin(1,b)=randi([1,12],1,1);
    end
  end
end
for origin(1,b)=randi([13,14],1,1);
end
end

tons_resi(:,1:c)=2*0.01.*pref_path_tons(:,1:c);
field_resi=randi([1,10],1,12);
intermediate_warehouse=[0 15 16 17];
biorefinery=[20 20 20 20];
pref_path1=pref_path(:,(2:13));
pp1=1:4:45;
pp2=2:4:46;
pp3=3:4:47;
pp4=4:4:48;
for h=1:c
    if any(pref_path1(bees,h)==pp1)==1||h==3||h==4||h==5||h==6
        intermediate_warehouse=0;
        trans_type1=0;
        trans_type2=transportation(forigin(1,h),20);
    elseif any(pref_path1(bees,h)==pp2)==1
        intermediate_warehouse=15;
        trans_type1=transportation(forigin(1,h),intermediate_warehouse);
        trans_type2=transportation(intermediate_warehouse,20);
    elseif any(pref_path1(bees,h)==pp3)==1
        intermediate_warehouse=16;
        trans_type1=transportation(forigin(1,h),intermediate_warehouse);
        trans_type2=transportation(intermediate_warehouse,20);
    else
        intermediate_warehouse=17;
        trans_type1=transportation(forigin(1,h),intermediate_warehouse);
        trans_type2=transportation(intermediate_warehouse,20);
    end

trans_type_resi=transportation(20$field_resi(1,h));

if h==1
    month1=[intermediate_warehouse; trans_type1; trans_type2; trans_type_resi];
elseif h==2
    month2=[intermediate_warehouse; trans_type1; trans_type2; trans_typeperesi];
elseif h==3
    month3=[0; trans_type1; trans_type2; trans_typeperesi];
elseif h==4
    month4=[0; trans_type1; trans_type2; trans_typeperesi];
elseif h==5
    month5=[0; trans_type1; trans_type2; trans_typeperesi];
elseif h==6
    month6=[0; trans_type1; trans_type2; trans_typeperesi];
elseif h==7
    month7=[intermediate_warehouse; trans_type1; trans_type2; trans_typeperesi];
elseif h==8
    month8=[intermediate_warehouse; trans_type1; trans_type2; trans_typeperesi];
elseif h==9
    month9=[intermediate_warehouse; trans_type1; trans_type2; trans_typeperesi];
elseif h==10
    month10=[intermediate_warehouse; trans_type1; trans_type2; trans_typeperesi];
elseif h==11
    month11=[intermediate_warehouse; trans_type1; trans_type2; trans_typeperesi];
else
    month12=[intermediate_warehouse; trans_type1; trans_type2; trans_typeperesi];
end
end
transportation_month(1:4,1:12)=[month1(:,1), month2(:,1), month3(:,1), month4(:,1), month5(:,1), month6(:,1), month7(:,1), month8(:,1), month9(:,1), month10(:,1), month11(:,1), month12(:,1)];
l=1;

for f=1:c
    if b_type(1,f)==1
        switchgrass(1,f)=pref_path_tons(1,f);
        stalk_straw(1,f)=0;
    else
        switchgrass(1,f)=0;
        stalk_straw(1,f)=pref_path_tons(1,f);
    end
end
end
number_units(1,1)=ceil(sum(switchgrass(1,:)).*2./8./7200);
cost_hu(1,1)=580000.*number_units(1,1);
hu_month(1,1)=cost_hu(1,1)/12;
harvest_unitcost(1,f)=hu_month(1,1).*f;

end
ii=4;
k=1;
for j=1:c
if j==1
    if transportation_month(1,j)==0
        processing_cost(k,j)=50*2*pref_path_tons(k,j);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j));
        revenue(k,j)=1.8*90*2*pref_path_tons(k,j);
    end
    if pref_path_tons(k,j)>60000
        inventory_cost(1,k)=2*(120000+tons_resi(k,j));
    else
        inventory_cost(1,k)=2*(2*pref_path_tons(k,j)+tons_resi(k,j));
    end
    if transportation_month(3,l)==1
        if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j));
            train_trans(1,k)=0;
        else
            truck_trans(1,k)=0.4*tons_resi(k,j);
            train_trans(1,k)=0.04*tons_resi(k,j);
        end
        else
            if transportation_month(4,l)==2;
                truck_trans(1,k)=0;
                train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j));
            else
                truck_trans(1,k)=0.4*tons_resi(k,j);
            end
        end
    else
        if transportation_month(4,l)==2;
            truck_trans(1,k)=0;
            train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j));
        else
            truck_trans(1,k)=0.4*tons_resi(k,j);
        end
    end
end
\[ \text{train}\_\text{trans}(1,k) = 0.04 \times (2 \times \text{pref}\_\text{path}\_\text{tons}(k,j)) \]

end

end

\[ \text{wb}\_\text{operation}\_\text{cost}(1,k) = 30000 \times 2 + 1666666.66667; \]

\[ \text{profit}(1,k) = \text{revenue}(k,j) - \text{processing}\_\text{cost}(k,j) - \text{purchasing}\_\text{cost}(k,j) - \text{inventory}\_\text{cost}(1,k) - \text{truck}\_\text{trans}(1,k) - \text{train}\_\text{trans}(1,k) - \text{wb}\_\text{operation}\_\text{cost}(1,k) - \text{harvest}\_\text{unitcost}(k,j); \]

cost\_\text{matrix}(1:9,k) = [\text{revenue}(k,j); \text{processing}\_\text{cost}(k,j); \text{purchasing}\_\text{cost}(k,j); \text{harvest}\_\text{unitcost}(k,j); \text{inventory}\_\text{cost}(1,k); \text{truck}\_\text{trans}(1,k); \text{train}\_\text{trans}(1,k); \text{wb}\_\text{operation}\_\text{cost}(1,k); \text{profit}(1,k)];

\text{else}

\[ \text{processing}\_\text{cost}(k,j) = 50 \times 2 \times \text{pref}\_\text{path}\_\text{tons}(k,j); \]

\[ \text{purchasing}\_\text{cost}(k,j) = 2 \times (50 \times \text{switchgrass}(k,j) + 35 \times \text{stalk}\_\text{straw}(k,j)); \]

\[ \text{revenue}(k,j) = 1.8 \times 90 \times 2 \times \text{pref}\_\text{path}\_\text{tons}(k,j); \]

\text{if} \ \text{pref}\_\text{path}\_\text{tons}(k,j) > 60000

\[ \text{inventory}\_\text{cost}(1,k) = 2 \times (120000 + \text{tons}\_\text{resi}(k,j) + \text{pref}\_\text{path}\_\text{tons}(k,j) \times 2); \]

\text{else}

\[ \text{inventory}\_\text{cost}(1,k) = 2 \times (2 \times \text{pref}\_\text{path}\_\text{tons}(k,j) + \text{tons}\_\text{resi}(k,j) + \text{pref}\_\text{path}\_\text{tons}(k,j) \times 2); \]

end

\text{if} \ \text{transportation}\_\text{month}(2,l) == 1

\text{if} \ \text{transportation}\_\text{month}(3,l) == 1

\text{if} \ \text{transportation}\_\text{month}(4,l) == 1

\[ \text{truck}\_\text{trans}(1,k) = 0.4 \times (\text{tons}\_\text{resi}(k,j) + 4 \times \text{pref}\_\text{path}\_\text{tons}(k,j)); \]

\[ \text{train}\_\text{trans}(1,k) = 0; \]

\text{else}

\[ \text{truck}\_\text{trans}(1,k) = 0.4 \times (4 \times \text{pref}\_\text{path}\_\text{tons}(k,j)); \]

\[ \text{train}\_\text{trans}(1,k) = 0.04 \times \text{tons}\_\text{resi}(k,j); \]

end

\text{else}

\text{if} \ \text{transportation}\_\text{month}(4,l) == 1

\[ \text{truck}\_\text{trans}(1,k) = 0.4 \times (\text{tons}\_\text{resi}(k,j) + 2 \times \text{pref}\_\text{path}\_\text{tons}(k,j)); \]

\[ \text{train}\_\text{trans}(1,k) = 0.04 \times (2 \times \text{pref}\_\text{path}\_\text{tons}(k,j)); \]

\text{else}

\[ \text{truck}\_\text{trans}(1,k) = 0.4 \times (2 \times \text{pref}\_\text{path}\_\text{tons}(k,j)); \]

\[ \text{train}\_\text{trans}(1,k) = 0.04 \times (\text{tons}\_\text{resi}(k,j) + 2 \times \text{pref}\_\text{path}\_\text{tons}(k,j)); \]

end
end
else
    if transportation_month(3,l)==1
        if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j));
            train_trans(1,k)=0.04*(2*pref_path_tons(k,j));
        else
            truck_trans(1,k)=0.4*(2*pref_path_tons(k,j));
            train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j));
        end
        train_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j));
    else
        if transportation_month(4,l)==1
            truck_trans(1,k)=0.4*(tons_resi(k,j));
            train_trans(1,k)=0.04*(4*pref_path_tons(k,j));
        else
            truck_trans(1,k)=0;
            train_trans(1,k)=0.04*(tons_resi(k,j)+4*pref_path_tons(k,j));
        end
    end
end

wb_operation_cost(1,k)=30000*2+1666666.66667+60000;
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
end
l=l+1;

best_option_costs1(1:9,j)=cost_matrix;
else
    processing_cost(k,j)=50*2*pref_path_tons(k,j)+processing_cost(k,j-1);
purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j))+purchasing_cost(k,j-1);
revenue(k,j)=1.8*90*2*pref_path_tons(k,j)+revenue(k,j-1);
if transportation_month(1,j)==0
    if pref_path_tons(k,j)>60000
        inventory_cost(1,k)=2*(120000+tons_resi(k,j))+best_option_costs1(5,j-1);
    else
        inventory_cost(1,k)=2*(2*pref_path_tons(k,j)+tons_resi(k,j))+best_option_costs1(5,j-1);
    end
end
if transportation_month(3,l)==1
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs1(6,j-1);
        train_trans(1,k)=0+best_option_costs1(7,j-1);
    else
        truck_trans(1,k)=0.4*(2*pref_path_tons(k,j))+best_option_costs1(6,j-1);
        train_trans(1,k)=0.04*tons_resi(k,j)+best_option_costs1(7,j-1);
    end
else
    if transportation_month(4,l)==2
        truck_trans(1,k)=0+best_option_costs1(6,j-1);
        train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs1(7,j-1);
    else
        truck_trans(1,k)=0.4*tons_resi(k,j)+best_option_costs1(6,j-1);
        train_trans(1,k)=0.04*(2*pref_path_tons(k,j))+best_option_costs1(7,j-1);
    end
end
wb_operation_cost(1,k)=30000*2+1666666.66667+best_option_costs1(8,j-1);
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
else
    processing_cost(k,j)=50*2*pref_path_tons(k,j)+processing_cost(k,j-1);
    purchasing_cost(k,j)=2*(50*switchgrass(k,j)+35*stalk_straw(k,j)+purchasing_cost(k,j-1);
revenue(k,j)=1.8*90*2*pref\_path\_tons(k,j)+revenue(k,j-1);
if pref\_path\_tons(k,j)>60000

inventory\_cost(1,k)=2*(120000+tons\_resi(k,j)+pref\_path\_tons(k,j)*2)+best\_option\_costs1(5,j-1);
else

inventory\_cost(1,k)=2*(2*pref\_path\_tons(k,j)+tons\_resi(k,j)+pref\_path\_tons(k,j)*2)+best\_option\_costs1(5,j-1);
end
if transportation\_month(2,l)==1
if transportation\_month(3,l)==1
if transportation\_month(4,l)==1
truck\_trans(1,k)=0.4*(tons\_resi(k,j)+4*pref\_path\_tons(k,j))+best\_option\_costs1(6,j-1);
train\_trans(1,k)=0+best\_option\_costs1(7,j-1);
else
truck\_trans(1,k)=0.4*(4*pref\_path\_tons(k,j))+best\_option\_costs1(6,j-1);
train\_trans(1,k)=0.04*tons\_resi(k,j)+best\_option\_costs1(7,j-1);
end
else
if transportation\_month(4,l)==1

truck\_trans(1,k)=0.4*(tons\_resi(k,j)+2*pref\_path\_tons(k,j))+best\_option\_costs1(6,j-1);
train\_trans(1,k)=0.04*(2*pref\_path\_tons(k,j))+best\_option\_costs1(7,j-1);
end
else

if transportation\_month(3,l)==1
if transportation\_month(4,l)==1
truck\_trans(1,k)=0.4*(tons\_resi(k,j)+2*pref\_path\_tons(k,j))+best\_option\_costs1(6,j-1);
train\_trans(1,k)=0.04*(2*pref\_path\_tons(k,j))+best\_option\_costs1(7,j-1);
end
end
else

if transportation\_month(3,l)==1
if transportation\_month(4,l)==1

truck\_trans(1,k)=0.4*(tons\_resi(k,j)+2*pref\_path\_tons(k,j))+best\_option\_costs1(6,j-1);
train\_trans(1,k)=0.04*(2*pref\_path\_tons(k,j))+best\_option\_costs1(7,j-1);
else
    truck_trans(1,k)=0.4*(2*pref_path_tons(k,j))+best_option_costs1(6,j-1);

    train_trans(1,k)=0.04*(tons_resi(k,j)+2*pref_path_tons(k,j))+best_option_costs1(7,j-1);
end

else
    if transportation_month(4,l)==1
        truck_trans(1,k)=0.4*(tons_resi(k,j))+best_option_costs1(6,j-1);
        train_trans(1,k)=0.04*(4*pref_path_tons(k,j))+best_option_costs1(7,j-1);
    else
        truck_trans(1,k)=0+best_option_costs1(6,j-1);
    end
end

wb_operation_cost(1,k)=30000*2+1666666.66667+60000+best_option_costs1(8,j-1);
profit(1,k)=revenue(k,j)-processing_cost(k,j)-purchasing_cost(k,j)-inventory_cost(1,k)-
           truck_trans(1,k)-train_trans(1,k)-wb_operation_cost(1,k)-harvest_unitcost(k,j);

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_unitcost(k,j);inventory_cost(1,k);
                    truck_trans(1,k);train_trans(1,k);wb_operation_cost(1,k);profit(1,k)];
end
l=l+1;
best_option_costs1(1:9,j)=cost_matrix;
end
end
pref_path_tons1(bees,1:12)=pref_path_tons;
hprofit1(bees,1)=best_option_costs1(9,12);
else
    pref_path_tons1(bees,1:12)=pref_path_tons2(bees,1:12);
end
clearvars -except tons tons_final tons_resi_all paths_profit_all best_option_cost_all forigin_all transportation_month_all nwaggedur1 v best_profit_bees best_profit_bees1 hprofit1 pref_path_tons2 c k hprofit pref_path1 pref_path_tons1 profit_bees tons_path path nbees lambda c months transportation alpha beta hive pref_path iterations
for bees=1:nbees
    if profit_bees(bees,1)>hprofit1(bees,1)
        nwaggle(bees,1)=bees;
    else
        nwaggle(bees,1)=0;
    end
end
nwaggle1=nwaggle;
nwaggle1(nwaggle1==0)=[];
is3=isempty(nwaggle1);
if is3==1
    pfollow=zeros(size(nwaggle,1),1);
else
    for zx=1:size(nwaggle1,1)
        nwaggedur(zx,1:25)=[path(nwaggle1(zx,1,:)) profit_bees(nwaggle1(zx,1,:))];
    end
    nwaggedur=[nwaggedur zeros(size(nwaggedur,1),1)];
    nwagglecum=zeros(size(nwaggedur,1)+size(nwaggedur,1),26);
    nwagglecum=[nwaggedur1;nwaggedur];
    for r=1:size(nwagglecum,1)
        if nwaggle(r,:)==0
            pfbee(r,1)=0;
        else
            pfbee(r,1)=nwagglecum(r,25);
        end
    end
end
pfcolony=(1/size(nwaggle1,1))*sum(pfbee);
for tpr=1:size(nwaggedur,1)
    if nwaggle(tpr,1)==0
        %
    end
end
pfcolony=(1/size(nwaggle1,1))*sum(pfbee);
for tpr=1:size(nwaggedur,1)
    if nwaggle(tpr,1)==0
        %
    end
end
% duration(tpr,1)=0
% else
duration(tpr,1)=k*(pfbee((size(nwaggledur1,1)+1,:),:)/pfcolony);
% end
duration1=duration;
duration1(duration1==0)=[ ];
duration1=ceil(duration1);
nwaggledur(:,26)=[duration1];
nwagglecum((size(nwaggledur1,1)+1):size(nwagglecum,1),:)=nwaggledur;
nwaggledur1=nwagglecum;
for xc=1:(size(nwaggledur1,1)-size(nwaggledur,1))
    nwaggledur1(xc,26)=nwaggledur1(xc,26)-1;
end
[ro2 co2]=find(0==nwaggledur1(:,26));
ro2=ro2';
nwaggledur1(ro2,:)=[];
for rs=1:size(nwaggledur1,1)
    if pfbee(rs,:)<.95*pfcolony
        pfollow(rs,:)=.6;
    elseif .95*pfcolony<pfbee(rs,:)<.975*pfcolony
        pfollow(rs,:)=.2;
    elseif .975*pfcolony<pfbee(rs,:)<.99*pfcolony
        pfollow(rs,:)=.02;
    else
        pfollow(rs,:)=0;
    end
end
pref_path_tons2=pref_path_tons1;
best_profit_bees(v,1)=max(profit_bees);
clearvars -except tons_final paths_profit_all best_option_cost_all forigin_all transportation_month_all
nwaggledur1 v best_profit_bees best_profit_bees1 hprofit1 c pfollow pbee duartion1 nwaggle nwaggle1 k hprofit
pref_path1 pref_path_tons1 profit_bees tons_path path nbees lambda c months transportation alpha beta hive iterations pref_path_tons2

end

best_profit_bees2=[best_profit_bees1; best_profit_bees];
plot(best_profit_bees2);
VITA

Jesusita Ibarra was born in Los Angeles, California. After graduating from the Centro Preparatoriano Computacional in Cuidad Juarez, Mexico in 1999 she entered the El Paso Community College. In the El Paso Community College she obtained an Associate’s Degree in 2007. She entered the University of Texas at El Paso that same year; where she received a bachelor’s degree in Industrial Engineering from the University of Texas at El Paso in fall 2010. Then, in fall of 2011, she started studying for her Master’s degree in Industrial Engineering. During that time she started working under the supervision of her advisor Dr. Heidi Taboada as a Research Assistant. In the spring semester of the 2013, she had the opportunity to assist to the ISERC Annual Conference held in San Juan, Puerto Rico where a research paper where she appears as a co-author was presented and published. Her research area is about the Biomass Feedstock Logistic System Design.

Permanent address:   3014 East San Antonio
                     El Paso, TX, 79905

This thesis/dissertation was typed by Jesusita Ibarra.