2013-01-01

Design Of A Biomass-To-Biorefinery Logistics System Through Bio-Inspired Metaheuristic Optimization Considering Multiple Types Of Feedstocks

Isidoro Trueba
University of Texas at El Paso, itrueba@miners.utep.edu

DESIGN OF A BIOMASS-TO-BIOREFINERY LOGISTICS SYSTEM THROUGH BIO-INSPIRED METAHEURISTIC OPTIMIZATION CONSIDERING MULTIPLE TYPES OF FEEDSTOCKS

ISIDORO TRUEBA

Department of Industrial, Manufacturing & Systems Engineering

APPROVED:

Heidi Taboada, Ph.D., Chair

Jose Espiritu, Ph.D.

Juan Noveron, Ph.D.

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

Dedication

This thesis is dedicated to my wife, parents and friends, for their guidance and support.

DESIGN OF A BIOMASS-TO-BIOREFINERY LOGISTICS SYSTEM THROUGH BIO-INSPIRED METAHEURISTIC OPTIMIZATION CONSIDERING MULTIPLE TYPES OF FEEDSTOCKS

by

ISIDORO TRUEBA

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

May 2013

Abstract

Bioenergy has become an important alternative source of energy to alleviate the reliance on petroleum energy. Bioenergy offers significant potential to mitigate climate change by reducing life-cycle greenhouse gas emissions relative to fossil fuels. The Energy Independence and Security Act mandate the use of 21 billion gallons of advanced biofuels including 16 billion gallons of cellulosic biofuels by the year 2022. It is clear that Biomass can make a substantial contribution to supplying future energy demand in a sustainable way. However, the supply of sustainable energy is one of the main challenges that mankind will face over the coming decades. For instance, many logistical challenges will be faced in order to provide an efficient and reliable supply of quality feedstock to biorefineries. 700 million tons of biomass will be required to be sustainably delivered to biorefineries annually to meet the projected use of biofuels by the year of 2022. This thesis is motivated by the urgent need of advancing knowledge and understanding of the highly complex biofuel supply chain. While corn ethanol production has increased fast enough to keep up with the energy mandates, production of biofuels from different types of feedstocks has also been incremented. A number of pilot and demonstration scale advanced biofuel facilities have been set up, but commercial scale facilities are yet to become operational. Scaling up this new biofuel sector poses significant economic and logistical challenges for regional planners and biofuel entrepreneurs in terms of feedstock supply assurance, supply chain development, biorefinery establishment, and setting up transport, storage and distribution infrastructure. The literature also shows that the larger cost in the production of biomass to ethanol originates from the logistics operation therefore it is essential that an optimal logistics system is designed in order to keep low the costs of producing ethanol and make possible the shift from fossil fuels to biofuels. In many ways biomass is a unique renewable resource. It can be stored and transported relatively easily in contrast to renewable options such as wind and solar, which create intermittent electrical power that requires immediate consumption and a connection to the grid. This thesis presents two different models for the design

optimization of a biomass-to-biorefinery 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 ant colony optimization. Compared to rigorous mathematical optimization methods or iterative algorithms, metaheuristics do not guarantee that a global optimal solution can be found on some class of problems. Problems with similar characteristics to the one presented in this thesis have been previously solved using linear programming, integer programming and mixed integer programming methods. However, depending on the type of problem, these mathematical or complete methods might need exponential computation time in the worst-case. This often leads to computation times too high for practical purposes. Therefore, this thesis develops two types of metaheuristic approaches for the design optimization of a biomass-to-biorefinery logistics system considering multiple types of feedstocks and shows that metaheuristics are highly suitable to solve hard combinatorial optimization problems such as the one addressed in this research work.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Importance of biomass

About 36 percent of the total U.S. major energy consumption in the year 2011 came from petroleum, 71 percent was utilized for transportation, 23 percent for industrial feedstocks and 5 percent was used for space heating of commercial and residential buildings (Stowe, 2012). Figure 1 shows the primary energy consumption by source and sector for the year 2011. According to the graph only 9 percent of the energy consumption comes from renewable energy proving the country reliance on petroleum.

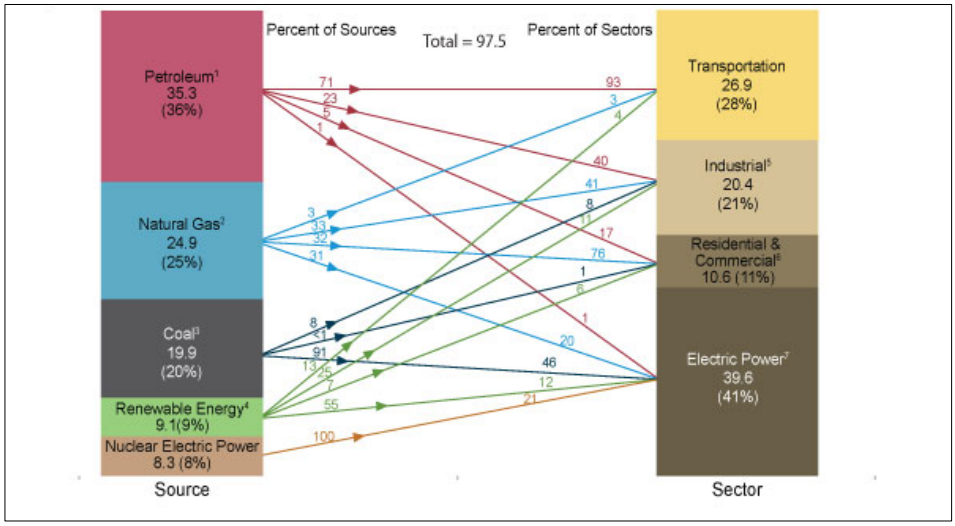


Figure 1: Primary Energy Consumption by Source and Sector

(U.S. Energy Information Administration)

From 1990 to 2011 the total U.S. greenhouse gas emissions have increased by 8.7 percent (figure 2), and a total of 6708.3 million metrics tons, CO_2 Eq. were generated in 2011, 86 percent of the green house gas emissions belong to the combustion of fossil fuels (inventory of U.S. Greenhouse Gas Emissions and sinks: 1990-2011).

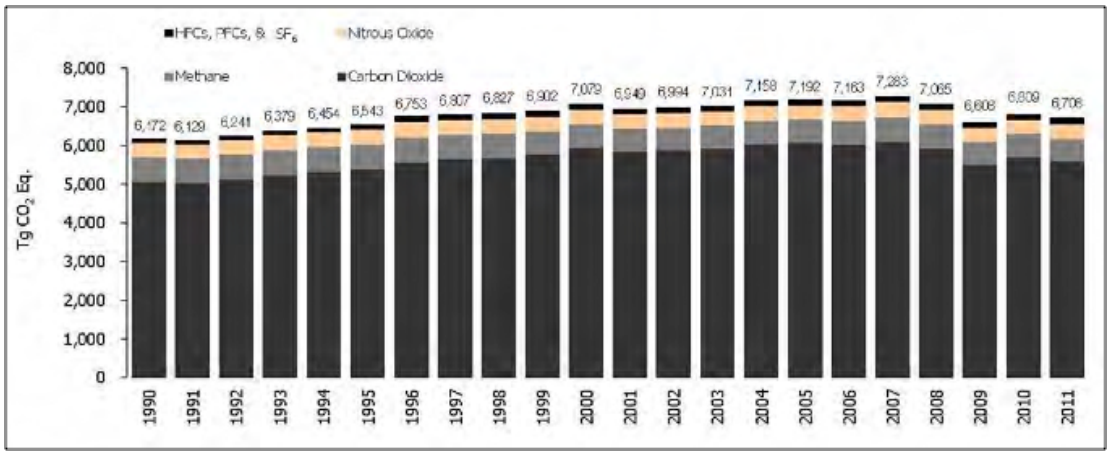


Figure 2: Greenhouse Gas Emissions by Gas

Also the reliance on petroleum energy occurs at an excessive price since the U.S. does not have the capacity of supplying their demand of petroleum. Figure 2 shows the oil consumption, production and imports to the U.S., and according to the graph, the level of consumption is greater than the oil produced in the United States and consequently the import of the mentioned product is very high in order to make up for that lack of local production.

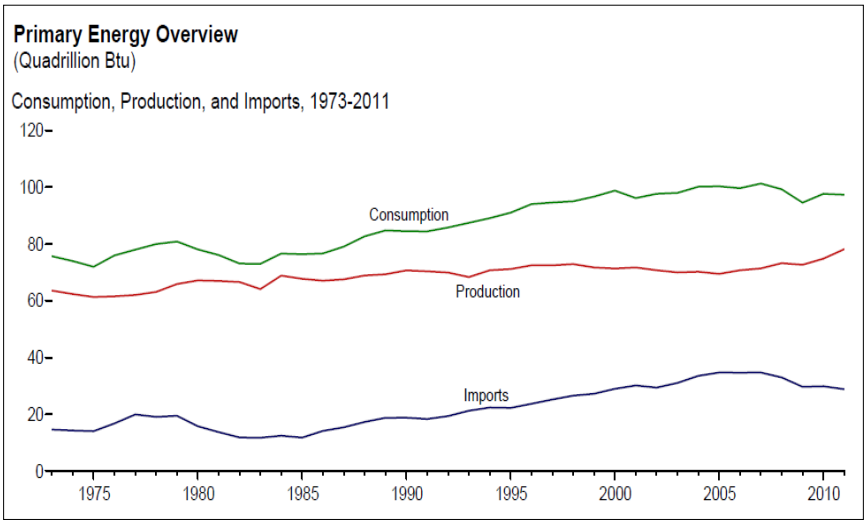


Figure 3: Energy Overview Source

(U.S. Energy Information Administration)

Over the years solutions have been introduced such as wind, solar, geothermal energy and Biomass to minimize the dependence on petroleum energy.

Biomass is a unique renewable resource that can be stored and transported relatively easily in contrast to renewable options such as the ones previously mentioned. Biomass is material contained in plants and it is derived from the reaction between CO_2 in the air, water and sunlight, via photosynthesis. Energy can be produced if biomass is efficiently processed. Biomass resources can be variable and it has four main categories (figure 4):

Wastes: agricultural production wastes, agricultural processing wastes, crop residues, mill wood wastes, urban wood wastes, and urban organic wastes.

Forest products: wood, logging residues, trees, shrubs and wood residues, sawdust

Energy crops: short rotation woody crops, herbaceous woody craps, grasses, starch crops, sugar crops, forage crops, oilseed crops.

Aquatic plants: algae, water weeds, water hyacinth, reed and rushes.

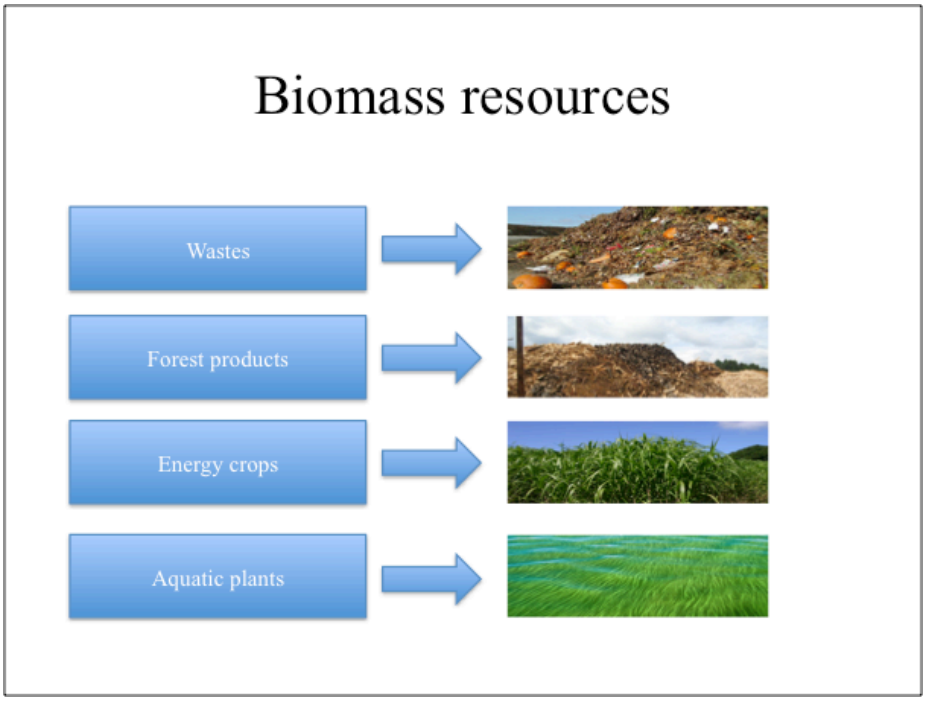


Figure 4: Biomass resources

Biomass has such a big potential that the U.S. is currently investing in developing alternative fuels such as biofuels. The Energy Independence and Security Act was enacted in 2007 and as part of it

36 billion gallons of ethanol per year are required to be produced by 2022 (Act, A., 2007). 21 out of the 36 billion gallons of ethanol must be produced from advanced biofuels (figure 5).

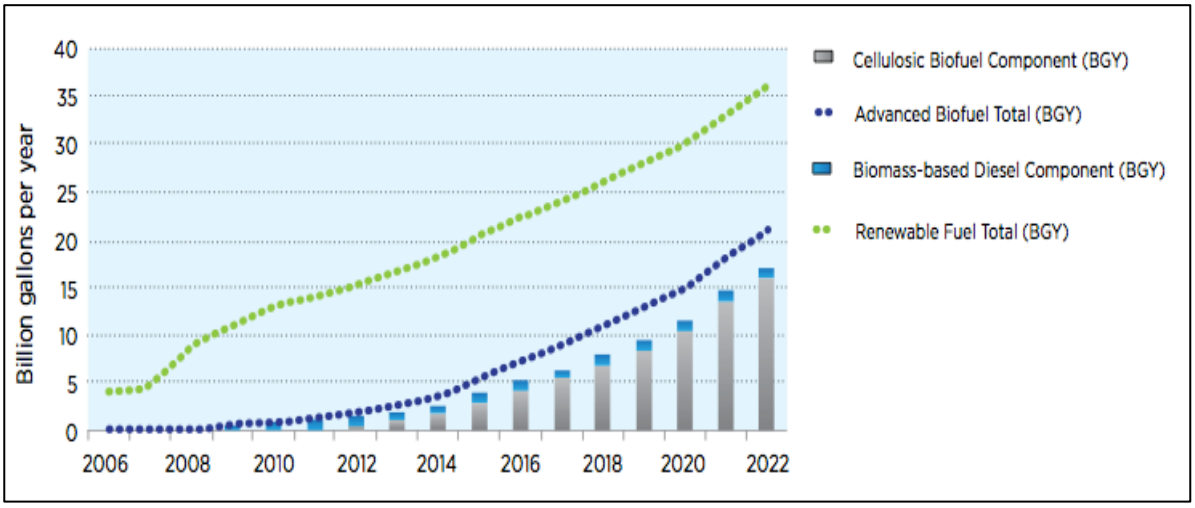


Figure 5: Energy Independence and Security Act (U.S. Department of Energy)

As of 2011 the total production capacity of the U.S. was 14.2 BGY as shown in figure 6. Most of the ethanol produced in order to satisfy the requirements by the Energy Independence and Security Act comes from corn since the production of ethanol from cellulosic and advanced fuels has become a sector with economical and logistical challenges for regional planers and biofuel entrepreneurs in the areas of feedstock supply, supply chain, biorefinery location, transportation, storage and distribution (Subbu and Satish, 2012). Advanced biofuels obtained from biomass such perennial grasses and agricultural residues can reduce the dependency in petroleum and contribute in the minimization of negative environmental impacts, improving the cost of living, and providing a good energy source but first the logistics system design of Biomass to Biofuels needs to be optimal so the cost of producing ethanol is minimized and consequently making in it more appealing to the investors.

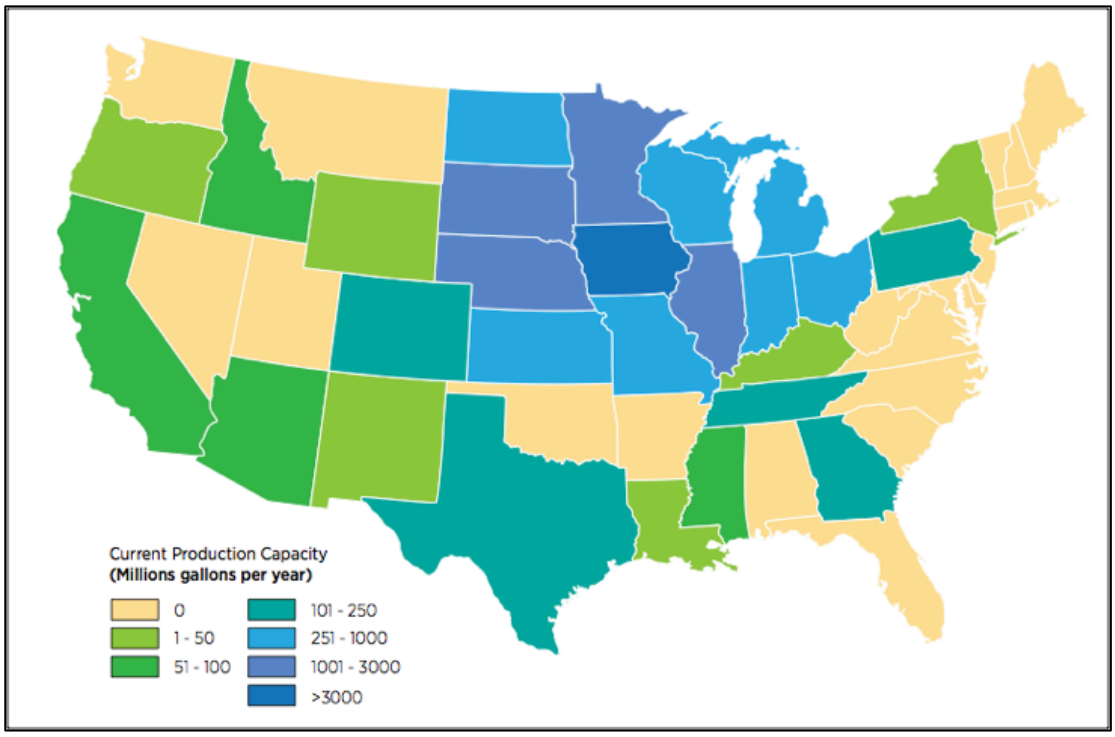


Figure 6: Ethanol production capacity by state (U.S. Department of Energy)

Problems involving the design of logistics system of biomass-to-biorefinery have been approached from a different perspective using optimization methods such as: Linear Programming (LP), Integer Programming (IP) and Mixed Integer Linear Programming (MILP). All of the optimization methods previously mentioned are deterministic algorithms that follow a rigorous procedure and its path and values of both design variables and the functions are repeatable. Deterministic Algorithms such as linear programming have many disadvantages such as being only applicable to problems where the constraints and objective function are linear meaning that when the constraints or objective functions are not linear these techniques cannot be used, also developing a deterministic algorithm can be time-consuming. A stochastic approach is introduced in this thesis in order to address the logistics system design problem and to show how likely a good meta-heuristic optimization method will provide a near-optimal solution (as good as a solution provide by any (LP), (MILP), (IP) optimization methods) in a reasonable computational time. A Genetic Algorithm (GA) and Ant Colony Optimization (ACO) algorithm are the two meta-heuristic optimization methods applied to the biomass-to-biorefinery

problem, which is something that to the best of knowledge has never been done before in this type of problem. At the end, a numerical example is presented in order to show the advantages of using meta-heuristic optimization methods such as the Genetic Algorithm (GA) and Ant Colony Optimization (ACO) algorithm to solve hard combinatorial optimization problems such as the one addressed in this research work.

1.2 Thesis Objective

This thesis proposes the use of two different bio-inspired algorithms (GA and ACO) in order to solve a logistics system design problem of a multi-commodity network flow structure, and demonstrate the advantages of meta-heuristic optimization methods over mathematical models.

The remaining of this thesis is grouped as follows. Chapter 2 provides literature review on the mathematical and heuristic approach to the logistics of biomass-to-biorefinery problem.

Chapter 3 will provide the problem statement and model framework of biomass-to-biorefinery containing the constraints and equations to optimize the system.

Chapter 4 describes the Genetic Algorithm (GA) and the Ant Colony Optimization Algorithm (ACO) approach for the problem described in Chapter 3.

Chapter 5 will provide a numerical example, computational results and sensitivity analysis of the application of the Genetic and Ant Colony Optimization algorithms.

Finally, chapter 6 presents conclusions and future research.

Chapter 2: Mathematical Techniques Approaches and Meta-Heuristic Optimization Methods

Chapter 2 will review several mathematical approaches for solving the biomass-to-biorefinery logistics problem and the description of several meta-heuristic methods that where contemplated to solve the biomass-to-biorefinery problem will be presented.

2.1 Mathematical Approaches

There is vast literature that approaches the optimization of biomass supply chains with mathematical models. In the next section literature approaching biomass supply chains problems with mathematical models such as Mixed Integer Linear Programming (MILP) and Linear Programming will be reviewed.

2.1.1 Linear and Mixed Integer Programming

In 2005 the U.S. energy policy was enacted and renewable fuel standards were established. The new renewable fuel standards required to double production of biofuels at present time by 2012, which was 7.5 billion gallons of biofuels that include ethanol and biodiesel. From 2005 to 2012 the production of cellulosic ethanol is to be counted with a 2.5 to one ratio and after 2012 the ratio will no longer apply. Minimum requirement of 250 million gallons of cellulosic biomass fuels are required by the RFS annually. Additionally to the energy act of 2005 several states implemented biomass incentives.

With the mandates and incentives of the energy act policy, ethanol production was accelerated at an unprecedented rate. In 2007 the Energy Independence act was established increasing the requirements to 36-Billion gallons of ethanol by 2022 and also requiring that 60 percent of the biofuels needed to be advanced fuels that cut green house emissions by at least 50 percent. Design of the logistics system is consider being one of the greatest challenges in the production of biofuels (Zhu et al. 2011).

Numerous works proposing mathematical models to optimize the design of the logistics system and improvements to the entire supply chain of biofuels can be found. Cundiff, et al. (1997) approaches

a problem with four different subsystems and the main goal is to design a biomass delivery system that considers transportation, storage, and scheduling issues. The authors of this article developed a mathematical model that is formulated as a linear program with two different weather scenarios were the transportation costs and the cost of capacity expansions at storage sites were to be minimized. Tembo et al. (2003) analyzed the challenges presented by the conversion of lignocellulosic biomass to ethanol developing a model, which is a multi-region, multi-period, mixed integer mathematical program that determines the most economical source of biomass in specific regions of Oklahoma.

A study was conducted to determine harvesting, storage and transportation costs from lignocellulosic biomass distribution system developing a multi-region, multi period-mixed integer mathematical program to approach it (Mapemba et al. 2004). A possible solution is presented to the problem encountered when trying to store biomass en-route to bio-energy plants were an integer programming formulation is utilized to solve this biomass location allocation problem. The main objective is to optimize the intermediate warehouses location and minimize transportation and storage location costs. A comparison of his study is made to one in which the intermediate warehouses are distributed uniformly over the area in between the farms, where the biomass is produced, and the bio-energy plants, in which the biomass is transformed to energy of some sort. The mathematical methods proposed by the author incurred in 79% of savings in comparison to the previously used method to establish warehouses (Judd et al. 2010). Xiaoyan et al. (2011) evaluated the problems and different circumstances that arrive while designing the logistics of a biomass-to-biofuel system. They proposed a Mixed Integer Linear Programming model. The research includes the planting, harvesting of the biomass, which, is switchgrass, as well as the delivering of the feedstock to the refinery, also deals with the disposal of the residue. As results of their research they reported that under a well-designed biomass logistics system, the mass production with a stable supply of biomass could increase the profit of bioenergy. Chen and Fan (2012) proposed a Mixed Integer stochastic programming model with the objective of supporting

strategic planning of bioenergy supply chain systems and optimal feedstock resource allocation in an uncertain decision environment. The study, focused on finding the optimal bioenergy supply infrastructure system design under steady-state parameters.

Also many mathematical models have been developed in order to address the multiple types of feedstock problem, since they have become necessary in the production of biofuels nowadays (Mitchell et al. 2010). Zhu and Yao (2011), proposed a mixed integer linear program where the main objective was to maximize profit on the complete logistics network for a system with three different types of biomass including their interaction in supplying the bioenergy production. You, et al. (2011), developed a multi-objective mixed integer linear program (MO-MILP) model to address the optimization of cellulosic ethanol supply chains using multiple types of biomass feedstock under economic, environmental and social objectives. Another mixed Integer linear programming (MILP) is developed to solve an optimization study that involves the net present value of a biomass to ethanol supply chain with five types of biomass residues which are converted by biochemical means into the biofuel Ethanol (Marvin et al. 2012). Shastri, et al. (2011), presented a study that deals with an integrated framework to connect various feedstock production related activities such as pre-harvest crop management, harvesting, transportation and pre-processing. A breadth level mathematical programming model called BioFeed is introduced as a Mixed Integer Linear Program (MILP.) The model has the main objective of determining the optimal configuration to the feedstock production system to maximize the total profit while incorporation long-term design decisions as well as management decisions. Bruglieri and Liberti (2008) addressed a problem that involves several processing plants, in which different types of crops are considered in order to obtain chemical energy. The model presented reflects the different types of costs involved in the system, such as transportation, processing and supply. A linear programming problem is developed that obtained a near-optimal solution that minimizes cost and maximizes profit.

2.2 Meta-Heuristic Optimization Methods

Meta-heuristics are considered by many results in literature as the state-of-art techniques for problems that do not have efficient algorithms (Cruz-Bernal 2013). According to Cruz-Bernal (2013) "The high increase in the size of the search space and the need of processing in real-time has motivated recent research's to solve problems using nature inspired heuristic techniques".

It has been proven that meta-heuristic have many desirable features to solve very complex problems such as supply chain management problem since they are easy to implement, and have been successful to solve difficult problems (Lourenco 2005). Natured inspired and mechanical procedures algorithms are contemplated in this research. A general description of nature inspired algorithms is shown in figure 7.

1. Initialize the solution vectors and all parameters.
2. Evaluate the candidate solutions.
3. Repeat
    a. Generation a new candidate solutions via the nature o social behaviors.
    b. Evaluate the new candidate solutions.
4. Until meet optimal criteria.

Figure 7: General Description for Nature Inspired Algorithms

Evaluation of the advantages and disadvantages of the algorithms are taken in consideration in order to determine which fits best in order to solve the problem of the biomass supply chain with multiple feedstocks.

Nature inspired algorithms considered are:

- Particle Swarm Optimization
- Bee Colony Optimization
- Firefly Algorithm
- Viral Systems

- Simulated Annealing
- Harmony Search
- Cuckoo Search Algorithm

2.2.1 Particle Swarm Optimization

Particle Swarm Optimization is a relatively recent optimization method that optimizes a problem by having a collection or swarm of different solutions, called particles, and moving them around a specified search space utilizing different formulas that will indicate their position and velocity within the space (Hassan, Cohanim, and de Weck 2004). Kennedy and Eberhart first introduced this method in 1995. They came up with this method by observing the behavior of two different natural organisms, fish and birds. When fish travel through the ocean, they move in large groups composed of many fish of same species, this behavior is called fish schooling.

It is done mostly for survival purposes and also helps fish to swim more easily since having another fish in front reduces the friction with the water. The survival factor comes in when a predator comes around. When the fish swim together in schools, they confuse predators and fish also know that predators are less likely to attack when they see large groups of fish compared to when they see a few fish swimming on their own. Young fish do not have a tendency to form schools, but as they grow older they start forming bigger and bigger groups. This suggests that this behavior is already on their genetic makeup and they just develop it later on in life.

Similarly, bird's travel together in flocks, which are groups composed of many birds. Just like fish, birds flock due to the fact that they become less vulnerable to predators if they belong to a large group of similar birds. There are several other reasons as to why birds exhibit this behavior. One of the most important of these reasons is foraging. Birds tend to form flocks while looking for food, which allows many birds to take advantage of the same food supplies or locate a food supply that a single bird had already found. Flocking also has a few disadvantages such as the increase in visibility and the

increase in risk of diseases spreading throughout the group (Mayntz 2003). Nevertheless, the advantages presented by flocking and schooling completely outweigh the disadvantages due to the fact that both fish and birds are social beings that feel more comfortable as part of a group of similar individuals.

PSO follows this pattern. The potential solutions known as particles fly through the problem space. Each particle is treated as a point in a d-dimensional space, which adjusts its own flying according to its flying experience as well as the flying experience of other particles. Figure 8 shows the concept of modification of a searching point by PSO.

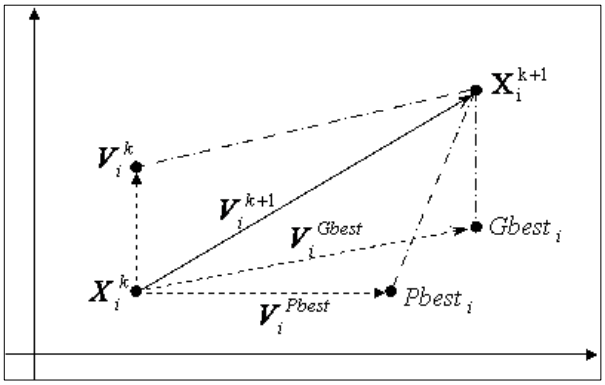


Figure 8: Concept of Modification of a Searching Point by PSO

X^k: current position

X^{k+1}: modified position

V^k: current velocity

V^{k+1}: modified velocity

$Pbest_i$: best previous position of *i*-th particle

$Gbest_i$: best particle among all particles

V^{Pbest}: velocity based on Pbest

V^{Gbest}: velocity based on Gbest

The main steps of the algorithm are shown in figure 9 which consist of five steps:

1. Initialization: Swarm population is generated, the position, velocity and local best are calculated for all the particles in the swarm, the gbest is set to be the local best
2. Update particles velocity:

3. Update position
4. Update bests: the fitness of each particle is evaluated according to the ne updated position
5. Stopping criteria: the process is repeated until the stopping criteria are met.

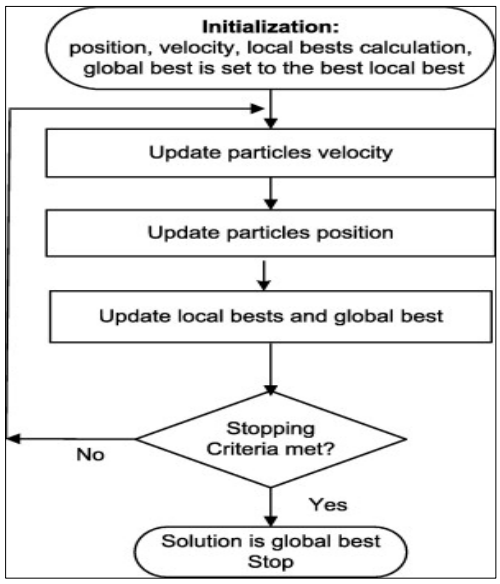


Figure 9: Flowchart of the PSO Algorithm

PSO techniques have been applied to solve TSP problems (Liu and Huang 2010), task allocation and knowledge workers scheduling (Qing and Han-Chao 2011), multi-agent based petroleum supply chain coordination (Ashesh et al. 2009), and robotics (Cruz-Bernal 2013).

PSO is considered to be a robust methodology that is easy to implement and only has a few parameters that need to be adjusted, considering these as advantages over evolutionary algorithms (EA) (Andras et al. 2012) and also can effectively solve a Supply Chain network optimization problem (Ashesh et al. 2009).

2.2.2 Bee Colony Optimization

Many social insect species are able to perform a variety of complex tasks; bees are one of those species. The process bees use to collect nectar it is highly efficient and very organized. Many algorithms

have been inspired by bees behavior, table 1 shows some of them and the problems that where studied with them.

Table 1: Algorithms Inspired By Bee's Behavior

Year	Authors	Algorithm	Problem studied
1996	Yonezawa and Kikuchi	Ecological Algorithm	Description of the collective intelligence based on bees' behavior
1997	Sato and Hagiwara	Bee System (BS)	Genetic Algorithm Improvement
2001	Lučić and Tedorović	BCO	Traveling Salesman Problem
2001	Abbas	MBO	Propositional satisfiability problems
2002	Lučić and Tedorović	BCO	Traveling Salesman Problem
2003	Lučić and Tedorović	BCO	Vehicle routing problem in the case of uncertain demand
2003	Lučić and Tedorović	BCO	Traveling Salesman Problem
2004	Wdde, Farooq, and Zhang	BeeHive	Routing protocols
2005	Tedorović, and Dell' Orco	BCO	Ride-matching problem
2005	Karaboga	ABC	Numerical optimization
2005	Drias, Sadeg, and Yahi	BSO	Maximum Weighted Satisfiability Problem
2005	Yang	Virtual Bee Algorithm (VBA)	Function optimizations with the application in engineering problems
2005	Benatchba, Admane, and Koudil	MBO	Max-Stat problem
2006	Tedorović, Lučić, Marković, and Dell' Orco	BCO	Traveling salesman problem and a routing problems in networks

Bee Colony Optimization (BCO) was first proposed as an optimization tool by (Lucic and Teodorovic 2001) for the Traveling Salesman Problem. Bee Colony Optimization (BCO) meta-heuristic has been introduced recently as a new way in the field of Swarm Intelligence (Lucic and Teodorovic 2003).

BCO is a population-based algorithm. A population of bees, which are artificial, searches for an optimal solution. This process contains several important steps. The first step in the foraging process of the bees consists of first leaving the hive in order to scout for food sources, the search these scout bees perform can extend to several kilometers in distance from the hive. As soon as the scout bee has been able to detect a good food source it returns to the hive. When all the scout bees are back in the hive with the information from where the food source is located an interesting act comes into the process, this step consists of the Waggle Dance performed by the bees (figure 10). This waggle dance contains critical information that is provided to their fellow bees. The information of the waggle dance contains distance, direction, food quality, the better the information the bee has to offer the longer the duration of the dance.

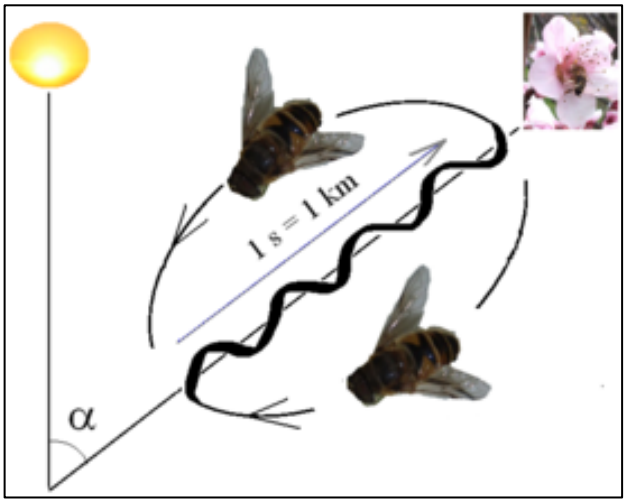


Figure 10: Waggle Dance

Once the bees watching the dance in the hive have analyzed the information, they decide either to use that information and follow (Figure 11 shows the recruiting of followers) that path or continue scouting for new locations containing new sources of food.

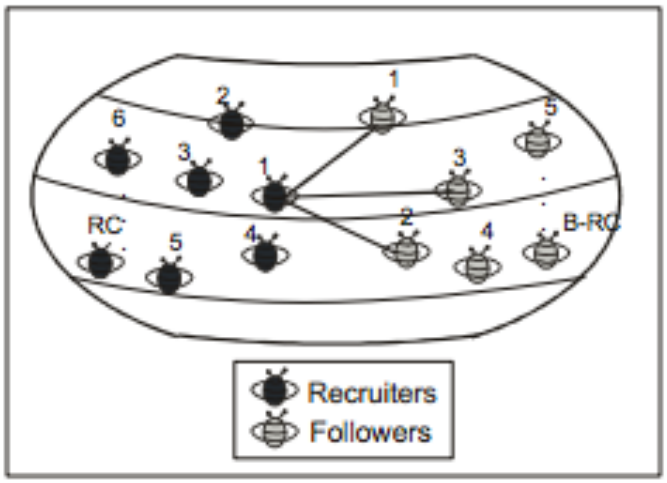


Figure 11: Recruiting of Followers

Next is represented the Pseudocode of the BCO algorithm (Lucic and Teodorovic 2003) :

1. Initialization: Bees are set to an empty solution;
2. For every bee do the forward pass:
    a. Set k=1
    b. Evaluate all possible moves
    c. Choose one move using roulette wheel;
    d. K=k+1
3. Every bee goes back to the hive
4. Evaluate bees and sort bees by their objective function value
5. Bees will decide if the become a recruiter or continue their own exploration
6. A solution from the recruiter will be assigned to every follower
7. Check stopping criterion, if not met go to step 2
8. Show the best results.

The algorithmic framework of BCO can be easily tailored to solve any type of optimization problems such as supply chain. Through the exchange of information and recruiting process, BCO has the capability to strengthen and exploit the regions of the solution space.

2.2.3 Firefly Algorithm

Fireflies emit a flashing light that is short and rhythmic; this flashing light is a way to communicate among other fireflies and also helps fireflies attract potential preys. This flashing light system used by fireflies was proposed and developed as an optimization tool by (Yang 2009; 2010) to solve complex problems. The Firefly Algorithm (FA) formulates the flashing light intensity in a way that associates with the objective function. Figure 12 shows the Firefly algorithm pseudocode.

For simplicity in the description of the FA three idealized rules are used (Yang, 2009; 2010):

1. Any firefly will be attracted to other flies. All fireflies are unisex
2. Attractiveness is proportional to the brightness, the less bright one will always move to the brighter one, if there is no brighter one the firefly will move random. As their distance increase their brightness will decrease.
3. The objective function will determine the brightness of the firefly.

Objective function $f(\mathbf{x})$, $\mathbf{x} = (x_1, ..., x_d)^T$
Generate initial population of fireflies $\mathbf{x}_i$ $(i = 1, 2, ..., n)$
Light intensity I_i *at* $\mathbf{x}_i$ *is determined by* $f(\mathbf{x}_i)$
Define light absorption coefficient γ
while *(t <MaxGeneration)*
for $i = 1 : n$ *all n fireflies*
  for $j = 1 : i$ *all n fireflies*
    if $(I_j > I_i)$, *Move firefly i towards j in d-dimension;* **end if**
    Attractiveness varies with distance r via $\exp[-\gamma r]$
    Evaluate new solutions and update light intensity
  end for j
end for i
Rank the fireflies and find the current best
end while
Postprocess results and visualization

Figure 12: Firefly Algorithm (FA) Pseudo code

Attractiveness, distance and movement need to be estimated in the firefly algorithm (FA). Equations 1, 2 & 3 define attractiveness, distance and movement respectively (Yang, 2009; 2010)

- Attractiveness:

$$\beta(r) = \beta_0 e^{-\gamma r^m}, \qquad (m \geq 1) \tag{1}$$

Where: r is the distance between two fireflies, β_0 is the initial attractiveness and γ is an absorption coefficient that controls the decrease light intensity.

- Distance:

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2}$$

Where: r is the distance between two fireflies x_i, y_i, x_j, y_j represent the Cartesian coordinates.

- Movement:

$$x_i = x_i + \beta_0 e^{-\gamma r^m}(x_i - x_j) + \alpha(rand - \frac{1}{2}) \tag{3}$$

Where: x_i represent the current position of the firefly, β_0 is the initial attractiveness and γ is an absorption coefficient that controls the decrease light intensity. α is the randomization parameter. The firefly algorithm (FA) has been applied in different areas such as power systems where (Sulaiman et al. 2012) implemented the Firefly Algorithm (FA) for solving an Economic Dispatch (ED) problem. Also has been applied in ship sailing path planning (Chang, Zhongqiang and Wheihua 2012). Firefly algorithm (FA) has shown advantages over Particle Swarm Optimization and Genetic Algorithms since it can deal with multimodal functions more efficiently. (Yang, 2009; 2010)

2.2.4 Viral Systems

Principals and processes of the vertebrates immune systems have inspired new computer algorithms, these new algorithms have the capability of exploiting the immune systems characteristics, which are learning, and solving problems and are known as artificial immune systems (AIS) (Cortes et al. 2007). A biological analogy based on viral infection was proposed by (Cortes et al. 2007) where they consider that the viruses are part of a general infection where every virus tries to act to its own benefit, which at the end results in the benefit of the viral system (VS).

The viral system (VS) has three components, which are a set of viruses, (figure 13 shows the simplest type of virus which is the phage), an organism and the interaction between them (Cortes et al. 2007). The virus in the VS is defined in four components: state, input, output and process. The second component of the viral systems consists of the organisms, which is defined by two components: state and process.

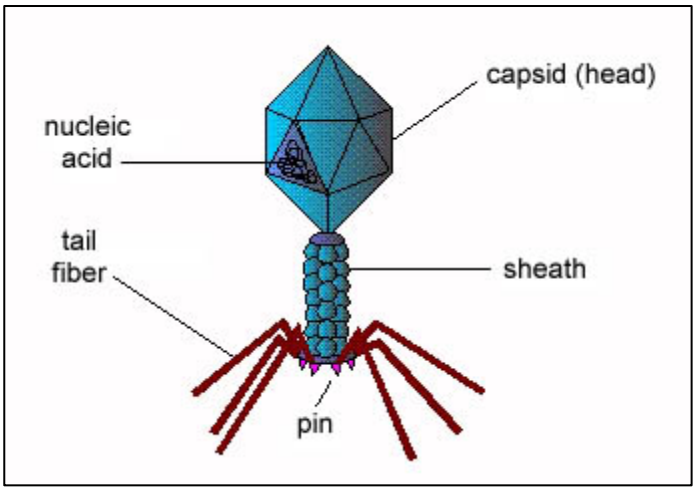


Figure 13: Coliphage Structure

The final component of the viral system is the interaction. This component is conditioned by the input and output actions that lead to a process of each and every virus and the response of the organism. Lytic and lysogenic replication mechanisms are capabilities that viruses have so they can weaken the immune response of any host. The replication process is represented in figure 14, the left side of the figure represents the evolution of the virus infection and it follows a series of steps that are explained next (Cortes et al. 2007):

1. The virus sticks to the border of the bacterium and after doing this it penetrates the border and introduces itself inside.
2. Once the virus is inside the cell stops its own production of proteins and it begins to produce proteins of the virus. The virus nucleus-capsids start replicating.

3. When several nucleus-capsids are replicated, the border of the bacterium breaks and the viruses are released and will infect the cells that are near.

Viruses lifecycle can be developed in one or more steps, some of them are capable of accommodating in cells which gives step to the lysogenic replication, this lysogenic replication is shown in the right side of figure 14 and it follows the next steps (Cortes et al. 2007):

1. The virus lodge in the genome by infecting the host cell.
2. By hiding in the cell the virus stays inside of it.
3. Lysogenic replication will produce a genome alteration and will lead to a similar mutation process.

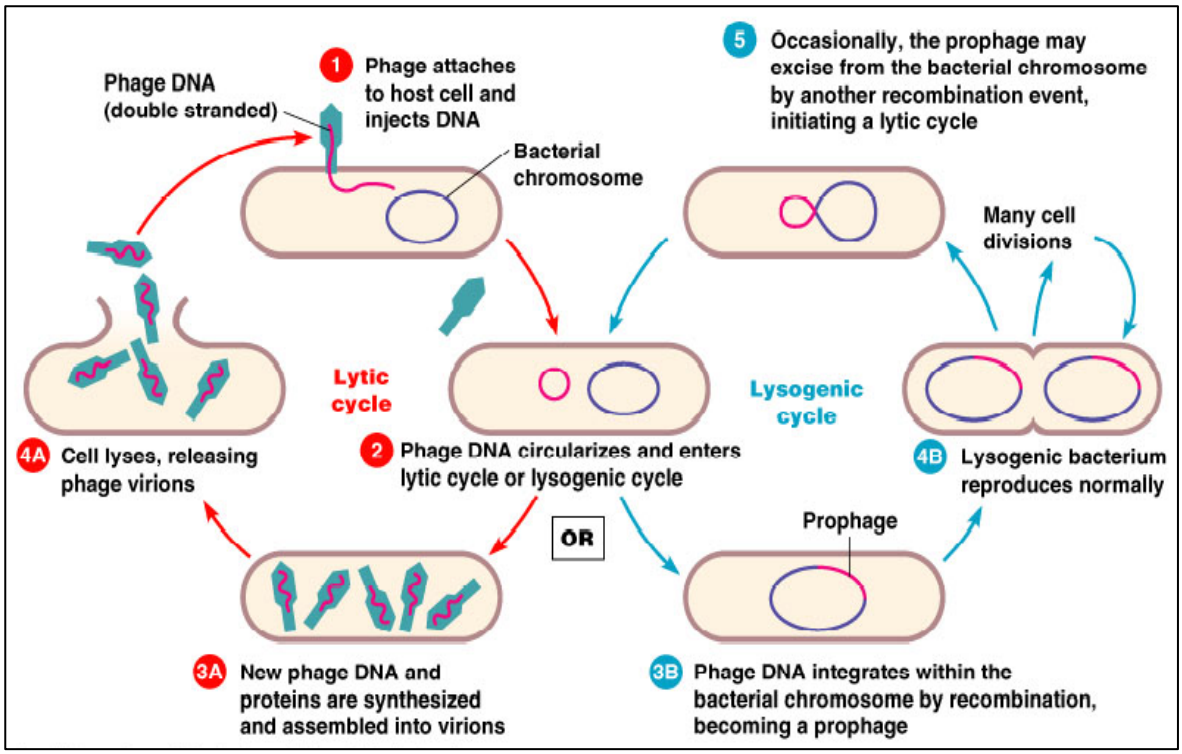


Figure 14: Lytic and lysogenic replication of viruses

In figure 15 a viral system flowchart is presented.

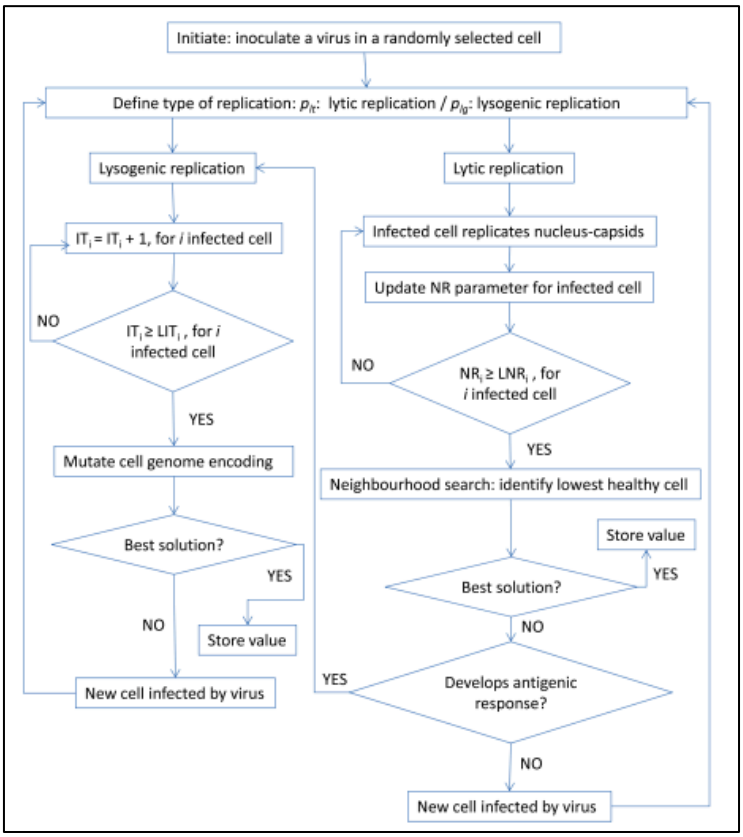


Figure 15: Viral System Flowchart

Little literature can be found where viral systems have been applied. (Cortes et al. 2007) applied viral systems (VS) to optimize the car dispatching in elevator group control systems by minimizing the waiting time of the passengers. Viral based optimization algorithm was used by (Espiritu et al. 2011) to find the optimal wind turbine placement where they consider constant wind speed and unidirectional uniform wind. It has also been utilized to solve knapsack problem (Suriyadi and Kusnadi 2011) and job scheduling problems (Cortes et al. 2010)

2.2.5 Simulated Annealing

Simulated annealing is an adaptation of the Metropolis-Hasting algorithm that was published by (Metropolis et al. 1953) and adapted by (Kirkpatrick et al. 1983). The optimization algorithm is based in a physical analogy annealing analogy that is a process in which a solid is heated until all particles are randomly arranged in a liquid state an then is followed by a slow cooling process (Busetti 2003). According to (Binder 2002) Simulated annealing involves three preparatory steps:

1. Analogies of the optimization and physical concept must be determined. Energy function equals to cost function, configuration of particles equals to the configuration of the problem parameters that is optimized and the temperature is equal to the control parameter of the optimization.
2. In order to define a decreasing set of temperatures an annealing schedule needs to be selected and the amount to spent at each temperature.
3. A way to generate and select new states must be defined.

Simulated annealing optimization takes place iteratively by first initializing randomly choosing points where cost is evaluated then the next new point gets chosen from a random number generator with a probability density where in case the cost of this point is better than the other point the new point is selected. In Each new iteration the probabilities for high deviations from the best point decrease. Figure 16 shows the flow chart of the simulated annealing algorithm.

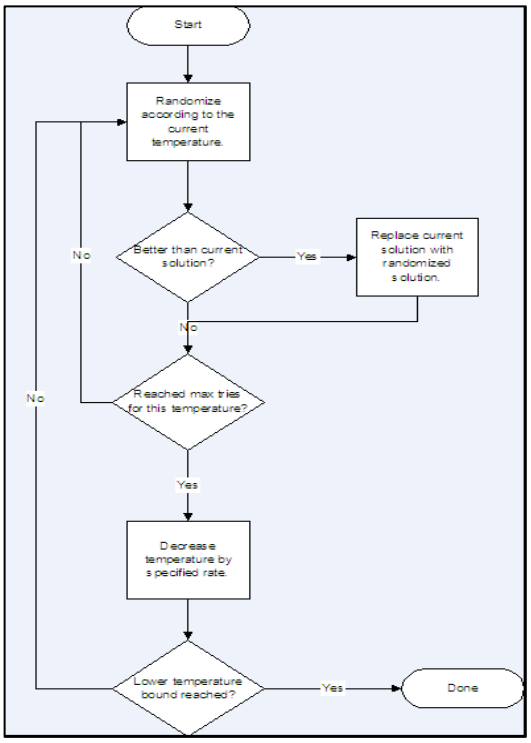


Figure 16: Simulated Annealing (SA) flowchart

Simulated annealing has been used in many supply chain problems as an approach to provide an optimal solution. For instance, (Ross and Jayaraman 2008) utilized simulated annealing to generate optimal distribution system design and utilization strategies. Heuristics algorithms were developed based on simulated annealing in order to find the optimal solution for the production quantity of material in each production lot and the job sequence with minimum supply chain total cost and lead time (Jung and Young 2009).

Simulated annealing can deal with non-linear models, noisy data and many constraints. It has a general technique, which gives an advantage of adapting the algorithm to any problem, but there is a tradeoff between the time required to get the solution and the quality of them.

2.2.6 Harmony Search

Harmony search is a metaheuristic algorithm developed by (Zoong et al. 2001). It has been applied to different optimization problems such as vehicle routing, water distribution networks, truss design, ground water modeling and many more. It was inspired by the search for a perfect state harmony. The quality of a musical instrument is determined by its frequency, sound quality, and the loudness. Sound quality also known as timbre is determined by the harmonic content which is determined by the waveforms of the sound signal. Harmonic that the instrument can generate will depend on the frequency range generated by the same (Yang 2009).

Harmony search it is based on the improvisation process of a musician, the process consist of three steps (Zoong et al. 2001):

1. Usage of harmony memory
2. Pitch adjusting
3. Randomization

The first step will ensure that the best harmony will be carried over to the new harmony memory. The parameter considering rate will be assigned to the usage memory so the memory can be used more effectively.

The second step which is the pitch adjusting and it will be determined by a pitch adjusting rate and also by a pitch band width. This step will be generating different solutions in the algorithm.

The final step that is randomization will increase the variety of solutions in the algorithm.

Figure 17 shows the pseudo code of the Harmony search algorithm

Harmony Search

begin

Objective function $f(\mathbf{x})$, $\mathbf{x}=(x_1,x_2,\ldots,x_d)^T$

Generate initial harmonics (real number arrays)

Define pitch adjusting rate (r_{pa}), pitch limits and bandwidth

Define harmony memory accepting rate (r_{accept})

while *(t<Max number of iterations)*

Generate new harmonics by accepting best harmonics

Adjust pitch to get new harmonics (solutions)

if *(rand>r_{accept}), choose an existing harmonic randomly*

else if *(rand>r_{pa}), adjust the pitch randomly within limits*

else *generate new harmonics via randomization*

end if

Accept the new harmonics (solutions) if better

end while

Find the current best solutions

end

Figure 17: Pseudo code Harmony Search algorithm (Yang 2009)

HS algorithm can be easily applied to different problems since the parameters are less sensitive to the chosen parameter meaning that the parameters don need to be adjusted in order to get good quality solutions.

Several approaches using Harmony search algorithm have been made in order to solve supply chain problems, (Purnomo, Wee and Praharsi 2012) solve a two inventory review policies on supply configuration problem were the objective was to minimize the sum of the inventory level and adds values by applying a Harmony search algorithm. To the best of knowledge Harmony search algorithm has not been applied to the biomass-to-biorefinery problem. Good opportunities can come from applying this metaheuristic algorithm to the problem proposed in this research.

2.2.7 Cuckoo search

Developed by (Yang and Deb 2009) and it is based on the obligate brood behavior of some cuckoo species and the combination of Levy flights of some other species such as fruit flies and birds.

A number of species engage the brood parasitism by laying their eggs in nest of other cuckoo species. Three types of parasitism can be found (Yang and Deb 2009):

1. Intraspecific brood parasitism.
2. Cooperative breeding
3. Nest takeover

A cuckoo bird can engage direct conflict with the intruder, also if a host discovers eggs that do not belong to the nest it will throw them away or leave the nest and build a new one.

In order to simplify the description of cuckoo search, three idealized rules are followed (Yang and Deb 2009):

1. Cuckoo birds will lay the eggs one at a time, and the nest will be selected randomly
2. The nests with the best eggs will carry over to the following generations
3. Available host nests will be fixed and the eggs that are laid by a cuckoo can be find by the host bird with a probability $P_a \in [0,1]$, where the host bird can throw the egg or abandon the nest.

Cuckoo Search via Lévy Flights

begin
  Objective function $f(\mathbf{x})$, $\mathbf{x} = (x_1, ..., x_d)^T$
  Generate initial population of
    n host nests $\mathbf{x}_i$ $(i = 1, 2, ..., n)$
  while *(t <MaxGeneration) or (stop criterion)*
    Get a cuckoo randomly by Lévy flights
      evaluate its quality/fitness F_i
    Choose a nest among n (say, j) randomly
    if $(F_i > F_j)$,
      replace j by the new solution;
    end
    A fraction (p_a) *of worse nests*
      are abandoned and new ones are built;
    Keep the best solutions
      (or nests with quality solutions);
    Rank the solutions and find the current best
  end while
  Postprocess results and visualization
end

Figure 18: Pseudo Code Cuckoo Search algorithm

In figure 18 the pseudo code for the Cuckoo search is shown. The algorithm starts by generating a initial population (first generation) of host nests, each nest will carry an egg that will determine a solution for the problem. After each generation only the best eggs will follow to the new generation. Quality of eggs will be improved by randomly selecting a new egg from the nest. If the egg selected randomly selected is better than the current egg also randomly selected from another nest, then the new egg will substitute the old egg. As mentioned previously with probability P_a a new egg will be randomly generated and it will replace the egg that is ranked the lowest in the nest's.

No approaches to the supply chain problems have been made using cuckoo search algorithm being this one of the disadvantages over the other metaheuristic optimization tool.

Particle Swarm Optimization (PSO), Bee Colony Optimization (BCO), Firefly Algorithm (FA), Viral Systems (VS), Simulated Annealing (SA), Harmony Search (HS) and Cuckoo Search (CS) are some of the meta-heuristic methods that were reviewed and considered in this chapter to approach the Biomass-to-biorefinery presented in this thesis. Genetic Algorithm (GA) and Ant Colony Optimization (ACO) algorithms were selected to approach this problem. The Biomass-to-Biorefinery problem presented in this thesis will be described in chapter 3.

Chapter 3: Biomass-to-Biorefinery Logistics Design

3.1 Problem Description

With the Renewable Fuel Standards established by the Energy Act policy in 2005 and the Energy Independence and Security Act in 2007, production of biofuels increase considerably but also the costs and challenges related to it. As mentioned previously one of the main issues in the production of biofuels is the design of the biomass-to-bioenergy logistics system that includes transportation network, feedstock supply, residue handling and distribution and the tactical operation schedules (Zhu and Qingzhu 2011).

This research is based on a logistics generic model designed by (Zhu and Qingzhu 2011). It is a multi-commodity network flow model that is a Mixed Integer Programming model that makes simultaneously strategic decisions and tactical schedules. The strategic decisions are for long-term systems-plans that are not subject to change in the near future (Hopp and Spearman 2001). These decisions include the distribution strategy of using transshipments via intermediate warehouses or direct transportation, composition of the harvesting team and the locations and capacities of the warehouses. Tactical schedules are the short-term decisions, which are the types and amounts of biomass that are harvested, purchased, stored and processed in each month.

A Genetic Algorithm (GA) and Ant Colony Optimization (ACO) algorithm are used to compare results against the mathematical model that was developed by (Zhu and Qingzhu, 2011) and determine the advantages of using meta-heuristic optimization methods to solve this type of problems.

3.2 Model

The Biomass-to Biorefinery logistics system that is being analyzed involves only two biorefineries that can be expanded to include more and three different types of biomass. Two types of transportation means have been selected that are train and truck. Production fields, intermediate warehouses and biorefineries compose all of the facilities and either train or truck can access them but

not necessarily both. Three types of feedstock are considered which are corn stalk, wheat straw and switchgrass. The production of switchgrass is the only feedstock that is included as part of the logistics system that is being analyzed. The remaining two that are wheat straw and corn stalk are produced outside the system and can be bought any time of the year. During the months of March, April, May and June switchgrass is not available since it is only harvested during the months of January, February, July, August, September, October, November and December. During the months that switchgrass is not available due to the harvesting season, wheat straw and corn straw will be bought in order to replace the lack of it. Wheat straw and corn stalk can only be introduced in the process the months that switchgrass is unavailable. Harvest units will be the ones conducting the harvest of switchgrass and are composed of:

- Harvest unit (Zhu and Qingzhu, 2011):
    - 10 Laborers
    - 9 Tractors
    - 3 Mowers
    - 3 rakes
    - 3 balers
    - 1 field transporter

As for storage of biomass several types of storage facilities are considered in the logistics system. The types of storage facilities considered are: in-field warehouse located at the switchgrass production field, intermediate warehouse and a warehouse located at each biorefinery. Switchgrass is going to be the only feedstock that is possible to store in the tree different types of storage facilities since is part of the system. Wheat straw and corn stalk are not going to be stored in either the in-field warehouse or the intermediate warehouse since they are outside the system, the only storage facilities that can be used are the ones located at each biorefinery. The residue that it is generated after the production of ethanol can be used as fertilizer for the fields, it will help in the preservation of the

qualities and minerals that are contained in the soil. By transporting back the residues to the fields will help preserve the sustainable structure in the logistics system of biomass-to-biorefinery. The transportation and residue is also taken into account in the model. Figure 19 shows the flow diagram of the biomass.

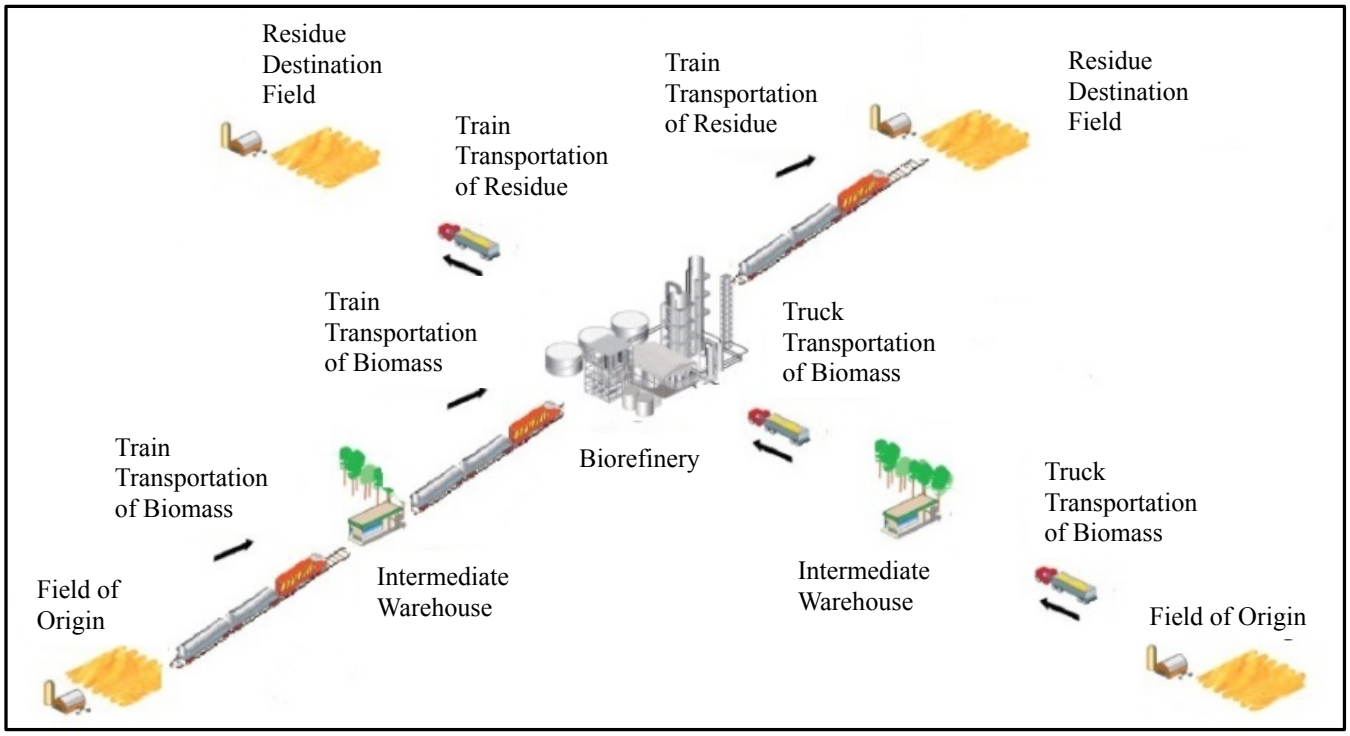


Figure 19: Biomass Flow Diagram adapted from USDA

3.3 Model Framework

The objective function for this logistics system is to maximize the total annual profit and is presented in equation 4:

$$\max R - \sum_{n=1}^{7} C_n \tag{4}$$

Where R is the revenue produced by the biofuel generating system

$$R = \sum_{m=1}^{12} \sum_{k=1}^{2} \sum_{l=1}^{3} \rho_m b_{mkl} \tag{5}$$

The total annual cost, represented by $\sum_{n=1}^{7} C_n$, is composed by:

- Processing Cost

$$C_1 = \sum_{m=1}^{12} \sum_{k=1}^{2} \sum_{l=1}^{3} c_{mkl} p_{mkl} \quad (6)$$

Where p_{mkl} is the dry tons of biomass *l* processed at biorefinery *k* in month *m*;

- Feedstock Purchasing Cost

$$C_2 = \sum_{m=1}^{12} \left\{ m_1 \sum_{i=1}^{10} h_{im_1} + m_2 \left(\sum_{s=1}^{2} \sum_{k=1}^{2} t_{skm_1} + \sum_{s=1}^{2} \sum_{k=1}^{2} t_{skm_2} \right) + m_3 \left(\sum_{t=1}^{2} \sum_{k=1}^{2} t_{tkm_1} + \sum_{t=1}^{2} \sum_{k=1}^{2} t_{tkm_2} \right) \right\} \quad (7)$$

where h_{im1} is the dry tons of switchgrass harvested from field *I* in month *m,* t_{skm1} and t_{skm2} are the dry tons of biomass transported by truck and train, respectively, to biorefinery *k* in field from field *s* month *m*, and t_{tkm1} and t_{tkm2} are the dry tons of biomass transported by truck and train, respectively, to biorefinery *k* in from field *t* month *m*;

- Inventory Cost of Biomass and Residue

$$C_3 = \sum_{m=1}^{12} \sum_{j \in J} \left(\sum_{l=1}^{3} \alpha_{ljm} s_{ljm} + \alpha_{0\,jm} s_{0\,jm} \right) \quad (8)$$

where s_{ljm} is the dry tons of biomass *l* stored at warehouse *j* during month *m* and s_{0jm} is the dry tons of residue stored at warehouse *j* during month *m*;

- Transportation Cost by Trucks

$$C_4 = \sum_{m=1}^{12} \left\{ \begin{array}{l} \sum_{i=1}^{10} \sum_{j \in J} \beta_{ijm11} t_{ijm11} + \sum_{j \in J} \sum_{k=1}^{2} \beta_{jkm11} t_{jkm11} + \\ \sum_{i=1}^{10} \sum_{k=1}^{2} \beta_{ikm11} t_{ikm11} + \sum_{s=1}^{2} \sum_{k=1}^{2} \beta_{skm21} t_{skm21} + \\ \sum_{t=1}^{2} \sum_{k=1}^{2} \beta_{tkm31} t_{tkm31} + \sum_{k=1}^{2} \sum_{i=1}^{10} \beta_{kim01} t_{kim01} \end{array} \right\} \quad (9)$$

where t_{ijm11}, t_{jkm11}, t_{ikm11}, t_{skm21}, t_{tkm31}, and t_{kim01} are the dry tons of biomass (1-switchgrass, 2-stalk, 3-stalk, 0-residue) transported from field *i*, *s*, or *t* to biorefinery *k* or to warehouse *j* in month *m* by truck;

- Transportation Cost by Trains

$$C_5 = \sum_{m=1}^{12} \left\{ \begin{array}{l} \sum_{i=1}^{10} \sum_{j \in J} \beta_{ijm12} t_{ijm12} + \sum_{j \in J} \sum_{k=1}^{2} \beta_{jkm12} t_{jkm12} + \\ \sum_{i=1}^{10} \sum_{k=1}^{2} \beta_{ikm12} t_{ikm12} + \sum_{s=1}^{2} \sum_{k=1}^{2} \beta_{skm22} t_{skm22} + \\ \sum_{t=1}^{2} \sum_{k=1}^{2} \beta_{tkm32} t_{tkm32} + \sum_{k=1}^{2} \sum_{i=1}^{10} \beta_{kim02} t_{kim02} \end{array} \right\} \quad (10)$$

where t_{ijm12}, t_{jkm12}, t_{ikm12}, t_{skm22}, t_{tkm32}, and t_{kim02} are the dry tons of biomass (1-switchgrass, 2-stalk, 3-stalk, 0-residue) transported from field *i*, *s*, or *t* to biorefinery *k* or to warehouse *j* in month *m* by train;

- Operation Cost of Warehouses and Biorefineries

$$C_6 = \sum_{m=1}^{12} \sum_{j \in J} \mu_j y_{jm} + \sum_{k=1}^{2} v_k z_k \quad (11)$$

where y_{jm} is a binary variable equal to 1 is warehouse *j* is open in month *m* and 0 otherwise, and z_k is a binary variable equal to 1 if biorefinery *k* is open and 0 otherwise;

- Operation Cost of Harvest Units

$$C_7 = \gamma u \quad (12)$$

where *u* is the number of employed harvest units.

The following constraints are present in the model and need to be considered while calculating the optimal value of the objective function:

- Production capacity (13)

$$\sum_{l=1}^{3} p_{mkl} \leq BCAP_{km} z_k \quad (13)$$

This constraint provides an upper limit on the feedstock amount of all types of biomass that can be processed at each biorefinery in each month;

- Safe inventory level - minimum processed biomass feedstock (14)

$$\sum_{l=1}^{3} p_{mkl} \geq BCAP_{km} z_k \quad (14)$$

This constraint imposes a safe inventory level on the minimum processed biomass feedstock in each biorefinery to avoid unexpected interruptions of biomass supply and biofuel production;

- Storage capacity (15)

$$\sum_{l=1}^{3} s_{ljm} \geq SCAP_j y_{jm} \quad (15)$$

The storage capacity constraint on warehouses sets a limit on the biomass feedstock residue that can be stored at the different types of 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 – minimum stored biomass (16)

$$\sum_{l=1}^{3} s_{lkm} \geq \delta SCAP_k y_{km} \quad (16)$$

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.

3.4 Model Assumptions

Several assumptions have been taken in consideration that describe the characteristics of the multiple feedstock system (Zhu and Qingzhu, 2011):

- Two transportation modes are available, truck and train.
- All Residues from the two biorefineries is re-circulated to switchgrass fields.
- Corn straw and wheat straw are not responsibility of central management since they are side products
- Corn Straw and Wheat straw can be purchased anytime.
- Switch grass cannot be harvested from March to June as previously mentioned.

Chapter 4: Genetic Algorithm and Ant Colony Optimization Approach

4.1 Genetic Algorithm

Genetic algorithms are part of the Evolutionary Algorithms and as the name indicates they evolve in order to find optimal solutions. GA's are inspired by Darwin's theory of biological evolution and natural selection. A genetic algorithm selects high strength classifiers as parents, forming offspring by recombining components from the parent classifiers. (Holland et al. 1992). Figure 20 represent the structure of the Genetic Algorithm.

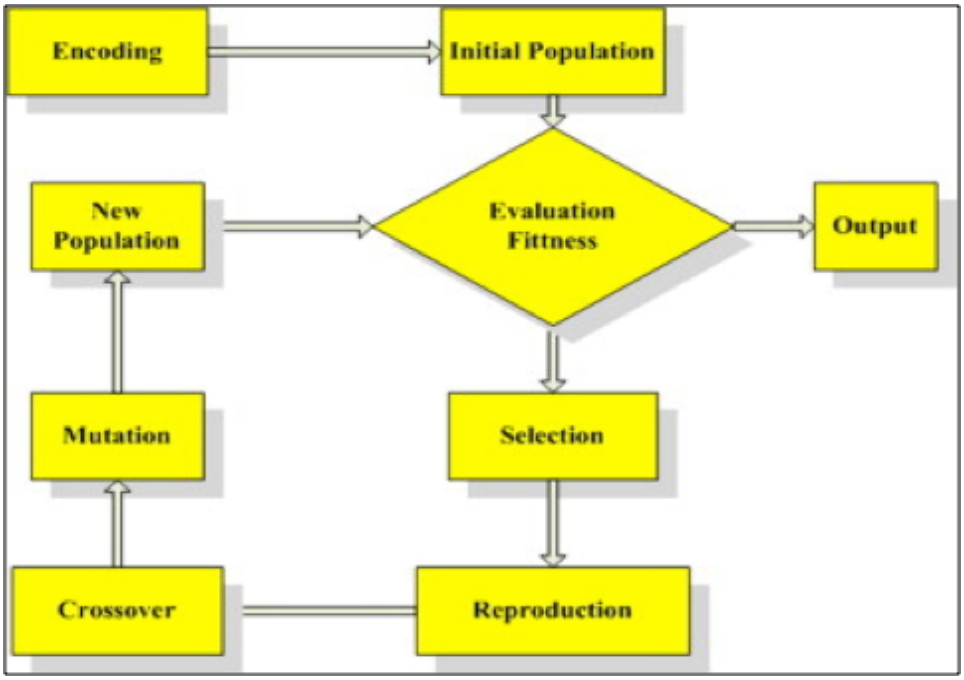


Figure 20: Genetic Algorithm Structure (Yun-Sheng et al. 2008)

The components of this algorithm are:

- Encoding technique
- Initialization procedure
- Evaluation function
- Selection of parents
- Genetic operators

- Parameter Setting

The encoding techniques are used to represent the chromosome in the problem solved, this techniques will be explained later in more detail. The initialization procedure will be the technique used in order to initialize the population. The evaluation function will be used in order to evaluate the fitness of the chromosomes in the population of chromosomes. Selection of parents is the technique utilized in order to select the parents from the population of chromosomes in order to reproduce and generate offspring's. Genetic operators will have the function of improving the offspring generated by the reproduction of the chromosomes selected to be the parents, a type of genetic operator is the mutation, this having the function to insert o replace one or more genes of the chromosome in order to obtain a different solution. Mutation operator needs to be applied in a very small percentage since having it applied in large percentage can lead to bad solutions.

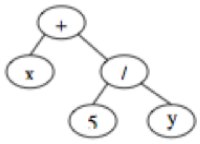
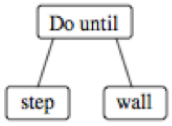
Also this algorithm adopts to its main steps and parameters some genetic terminology such as:

- Chromosome – genetic material contained in a string, in this case is a possible solution for the problem containing important information.
- Selection – chromosomes that will take part as parents in crossover.
- Mutation – inserting or replacing one or several genes in to the chromosome.
- Crossover – the mixing of the parents chromosomes to create new population.

4.1.1Encoding Techniques

In order to represent a chromosome encoding has to be implemented which is dependent upon the problem to be solved. Several encoding techniques are use such as binary, permutation and value encoding, these are represented in table 2 (Noor 2007).

Table 2: Encoding Techniques, (Noor 2007)

S/No.	Encoding technique	Example
1	Binary encoding	Chromosome 1 :1011001011001010111001 01 Chromosome 2 : 1111111000001100000111 11
2	Permutation encoding	Chromosome 1: 8 9 7 4 6 2 3 5 1 Chromosome 2 : 2 3 1 4 8 5 6 7 9
3	Value encoding	Chromosome 1: 5.3243 1.2324 2.3293 0.4556 2.4545 Chromosome 2 : HDIERJFDJDLDFABDJEIFLFEGT Chromosome 3 : (right), (back), (left), (back), (forward)
4	Tree encoding	Chromosome A  (+ x (/ 5 y)) Chromosome B  (do until step wall)

Each chromosome is a solution to the problem, and also pertains to a set of possible solutions that is called population, once the population is randomly generated; the chromosome is composed of its own unique genes.

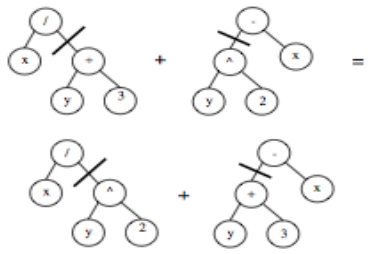
4.1.2 Selection Techniques

After performing encoding, generating an initial population and evaluating the next step in GA is selection, which will select the fittest parents that would take part in the crossover process. Several well-known techniques of selection are tournament selection, proportional roulette wheel selection and rank-based roulette wheel roulette (Mohd 2011).

4.1.3 Crossover Techniques

The crossover step is the process where is decided how combine the genes of the parents so children can be created. Crossover can be carried out in many different ways, in table 3 the most common type of crossovers are shown. Encoding will be a factor that it can help determine the type of crossover to be used.

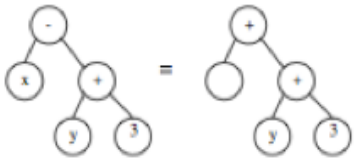
Table 3: Crossover Techniques (Noor 2007)

S/No.	Encoding technique	Crossover technique	Example
1	Binary	1. single cut point crossover	**1100/1011**+1101/1111= **1100**/1111 + 1101/**1011**
		2. Double cut point crossover	**11/00/1011** + 11/01/1111 = **11011011** +11001111
		3. Uniform crossover	**11001011** + 11011101 =**11011111** + **11001001**
		4. Arithmetic crossover	**0100110** + 0100101 = **1100**110 + **0101**111 **(the first three digits of each parent are added together for child 1 and the remaining digits are added for the child 2)**
2	Permutation encoding	Single cut point	**(1 2 3 4 5 6 /7 8 9) + (4 5 3 6 8 9/ 7 2 1) = (1 2 3 4 5 6 8 9 7) +4 5 3 6 8 9 7 1 2**
3	Real value encoding	Same as in binary values	**(1.29 5.68/ 2.86 4.11 5.55)**+(3.23 4.45/ 6.13 5.67 2.98) = **(1.29 5.68** 6.13 5.67 2.98) +(3.23 4.45 **2.86** 4.11 5.55)--**single point**
4	Tree encoding	Exchange	

4.1.4 Mutation Techniques

The next step after crossover in the GA is mutation, this process is carried out in order to induce into the population diversity, by doing this it will prevent the GA getting stuck into a local optimum (Obtiko 1998). Several mutation types can be used and it also depends in the encoding of the problem, in table 4 the most common are shown.

Table 4: Mutation Techniques (Noor 2007)

S/No.	Encoding type	Mutation technique	Example
1	Binary	Bit inversion	11001001 => 10001001
2	Permutation	Change in order	(1 **2** 3 4 5 6 **8** 9 7) => (1 **8** 3 4 5 6 **2** 9 7)
3	Real value	Addition of a small number	(1.29 2.86 5.68 **4.11** 5.55) => (1.29 2.86 5.68 **4.22** 5.55)
4	Tree	Change in operator	

According to Fotakis et al. (2012) GA's have an advantage over traditional methods in the solution of complex optimization problems, due to the fact that GA's differ fundamentally from traditional methods; utilizing coding of parameters, searching from a population, using objective function values and not derivatives and working with probabilistic operators.

4.2 Ant Colony Optimization

The ant colony optimization algorithm was invented by Marco Dorigo in 1990. It was inspired by real ant colony behavior instead of being based on the survival of the fittest such as in the Genetic Algorithm case. The foraging way of the ants motivated the elaboration of the ACO. The algorithm is particularly focused on how the ants find the shortest path to get their food. First of all, the ants start exploring the area around their nest in a random way to find the food.

When the ants are exploring different paths at the same time they are leaving a pheromone trail behind. The pheromone is a chemical substance that attracts and guides other ants to the food source. The trails that have a greater concentration of pheromone are the ones that most likely the ants will select. If the ant finds a good food source, it immediately identifies the quality of the food and carries it back to the nest. In the returning trip to the nest the ant leaves a quantity of pheromone and the amount

depends on the quality and quantity of food that the ant carries back to the nest. The indirect communication that exists between the ants is called stigmergy (Blum 2005).

Next, figure 21 shows an example of the ants foraging behavior that is described next:

Picture A - Indicates that all the ants are in the nest and that the paths do not contain pheromone.

Picture B - Represents when beginning of the foraging, here the ants have the same probability 50% of choosing either path. The ants that take the shorter path are represented in the figure as circles and the ones taking the longest path are symbolized as rhombs.

Picture C - Ants that have taken the shortest path will arrive first to the nest. Consequently, the probability of other ants selecting that same path again will be higher.

Picture D - To conclude since the pheromone evaporates in the longer path, the whole colony will exclusively use the short path.

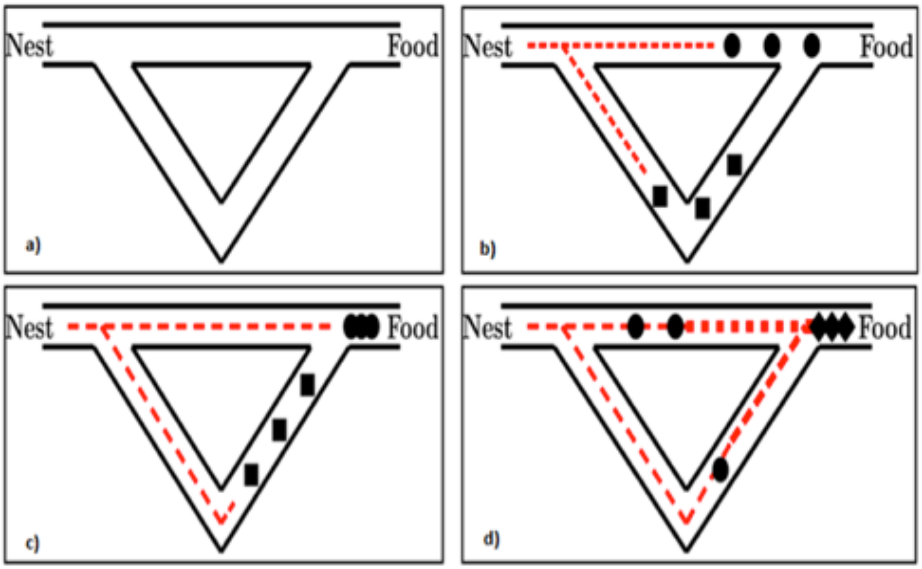


Figure 21: Ant Foraging Behavior

A flow chart is shown in figure 22 were the constructions of the solutions is done by using the objective function of the problem and a transition probability function (equation 17)

$$P_{ij} = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{l=1}^{a_i}(\tau_{il})^{\alpha}(\eta_{il})^{\beta}} & j \in AO \\ 0 \end{cases} \tag{17}$$

Where α, and β are the parameters that control the relative weight of the pheromone and the local heuristic. AO is the set of available options for the logistics system. All the decision variables previously presented will help determine the solution of each ant

The pheromone update will performed by using equation 18

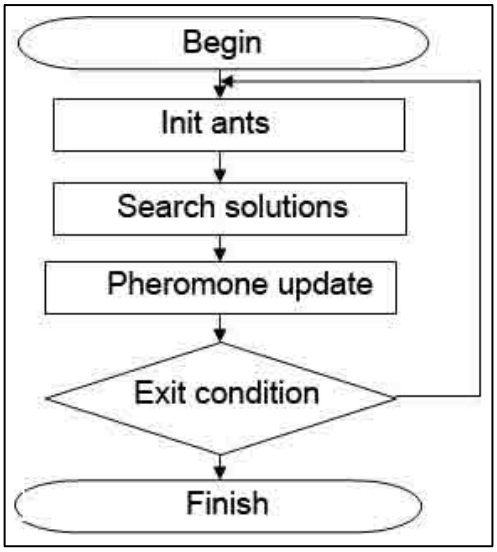


Figure 22: ACO Flow Diagram

$$\tau_{ij}^{new} = \rho \cdot \tau_{ij}^{old} + (1-\rho) \cdot \sum_{m=1}^{E}(E-m+1) \cdot P_m \tag{18}$$

where m = 1 will represent the best feasible solution found until now and the best E ants will be the ones that contribute their pheromone to the trail intensity where the magnitudes of contributions are weighted by their ranks in the colony. P_m represents the profit.

4.3 Model Development

4.3.1 Genetic Algorithm Model

The Genetic Algorithm one of the meta-heuristic optimization methods that are used to solve the problem and as its base was used the model presented in the previous section. In order to evaluate and determine the fitness of every chromosome the objective function and constraints that were previously presented are included in the algorithm. Every chromosome that did not fulfill the constraints included in the model it was automatically eliminated and therefore not included as a member of the population. The chromosome consisted of five subsections that represented the decisions variables presented by the model (table 5).

Table 5: Chromosome Description

Months	1	2	3	4	5	6	7	8	9	10	11	12	Processed Biomass
Biomass Type	1	1	2	3	2	3	1	1	1	1	1	1	
Field Origin	8	7	11	14	12	13	5	9	1	10	2	4	
Tons of biomass processed	45553	101567	33567	78901	54656	90867	70604	65511	31906	102637	61901	79055	
Biorefineries Open	3	3	3	3	3	3	3	3	3	3	3	3	
Tons of Residue Produced	911.06	2031.3	671.34	1578	1093.1	1817.3	1412.1	1310.2	638.12	2052.7	1238	1581.1	Recirculated Residue
Residue Recirculated to Field	5	10	2	8	3	9	1	7	6	1	10	3	
In-Field Warehouse	5	10	2	8	3	9	1	7	6	1	10	3	Residue Storage
Tons of Residue Storage	911.06	2031.3	671.34	1578	1093.1	1817.3	1412.1	1310.2	638.12	2052.7	1238	1581.1	
Types of Biomass Stored	4	4	4	4	4	4	4	4	4	4	4	4	
Biomass Type	1	1	2	3	3	2	1	1	1	1	1	1	Transported Biomass
From Field Origin	8	7	11	14	12	13	5	9	1	10	2	4	
Intermidiate Warehouse Opne	0	1	0	0	1	0	1	1	1	0	0	0	
Possible Intermediate Warehouse Open	0	0	16	16	0	0	0	0	0	0	0	0	
Transportation Type	1	2	2	2	2	1	2	1	1	1	2	1	
To Biorefinery	20	20	20	20	20	20	20	20	20	20	20	20	
Transportation Type	1	1	2	1	1	1	1	2	2	2	1	2	
Tons of Biomas Transported	45553	101567	33567	78901	54656	90867	70604	65511	31906	102637	61901	79055	
Biomass Type	4	4	4	4	4	4	4	4	4	4	4	4	Transported Residue
From Biorefinery	20	20	20	20	20	20	20	20	20	20	20	20	
To Switchgrass Field	5	10	2	8	3	9	1	7	6	1	10	3	
Transportation Type	1	1	1	1	1	1	1	2	1	1	1	1	
Tons of Residue Transported	911.06	2031.3	671.34	1578	1093.1	1817.3	1412.1	1310.2	638.12	2052.7	1238	1581.1	

4.3.1.1 Encoding Section 1 – Processed Biomass

Section one of the chromosome represents the following:

1. Month: Each gene represented the month in the year.
2. Biomass Type: Represented the type of biomass processed at biorefineries with values from 1 to 3, where:

a. 1 represented switchgrass

b. 2 represented corn stalk

c. 3 represented wheat straw

3. Field of origin: Field where biomass originated.
4. Tons of biomass processed: represented the quantity in tons that were processed during each month
5. Biorefineries open: Refinery open during each month represented by:

    a. 1 – Biorefinery 1 open

    b. 2 – Biorefinery 2 open

    c. 3 – Biorefinery 1 & 2 open

    In order to maximize profit and since each biorefinery could be open or closed during the whole year and could not be closed down during certain months it was decided to operate biorefineries throughout the year.

4.3.1.2 Encoding Section 2 &3-Recirculated Residue and Residue Storage

Section two of the chromosome represents the following:

1. Tons of residue produced: amount of residue recirculated monthly
2. Residue recirculated to field: destination field for biomass residue

In this section biomass type is not part of it since the residue becomes the same after the biomass process.

Section three represents the amount of residue that is being stored in each field:

1. In-field warehouse: storage facility located in field
2. Tons of residue storage: amount of residue being stored in the in-field warehouse
3. Type of biomass stored: 4 represented Biomass residue.

4.3.1.3 Encoding Section 4- Transported Biomass

Section four represents the biomass transportation flow:

1. Biomass type: biomass type transported
2. From field origin: biomass transportation field of origin
3. Intermediate warehouse open: utilization of intermediate warehouse where
    a. 1 represented intermediate warehouse open
    b. 2 represented intermediate warehouse closed
4. Transportation type: transportation type used to transport biomass from field of origin to intermediate warehouse if any, where
    a. 1 represented truck
    b. 2 represented train
5. To biorefinery: biomass end destination where
    a. 20 represented biorefinery one & two
6. Transportation type: transportation type used to transport biomass from field of origin or intermediate warehouse to biorefinery, where
    a. 1 represented truck
    b. 2 represented train
7. Tons of biomass transported: amount of biomass transported represented in tons

4.3.1.4 Encoding Section 5- Transported Residue

Section five represents the transportation of residue, where:

1. Biomass type: type of biomass transported, in this case is biomass residue and it is represented by the number 4
2. From biorefinery: origin of biomass residue
3. To switchgrass fields: biomass residue field destination

4. Transportation type: type of transportation utilized to move biomass residue from biorefinery to switchgrass field, where
    a. 1 represented truck
    b. 2 represented train
5. Tons of residue transported: amount of residue transported represented in tons

In total the chromosome consisted of twenty-three rows where not all of them where able to vary through different generations. Out of the twenty-three only three of them were able to do so. In each chromosome these three rows are:

- Row 4 – the number of tons that are processed in each biorefinery
- Row 7 – the production fields the residue was recirculated to
- Row 13 – utilization of intermediate warehouse in each month

Variability limitation in most of the rows was due to the fact that most of the rows in each chromosome are closely related to each other and cannot be interchangeable. Rows 4, 7 and 13 are the only sections of the chromosome that are part of the GA crossover step in order to maintain integrity of the chromosome.

4.3.1.5 GA Model Steps-Techniques and Parameter

A Genetic Algorithm was used to solve the presented problem. The GA follows the next steps in order to solve the objective function:

1. Rank Selection: selection method used for this algorithm in which the entire population of chromosomes is taken in consideration, evaluated and sorted from the best fitness value to the worst. Fitness values with higher value will be the ones with higher probability of being selected as a potential parent. Rank selection parameter value considered in this research is .7 % where only 70% of the best fitness values were considered as parents

2. Single Point Crossover: cross over method selected for this algorithm. This method selects two chromosomes as parents in order to reproduce and generate to children. For the first child a single random point will be taken in the chromosome and before this point will have genes from the firs parent and after the random point it will have genes from the second parent. For the second child the opposite will be done, the first part is going to have genes from the second parent and the second part will have genes from the first parent. The point that is generated randomly generally falls in the middle of the chromosome so symmetry of the chromosome can be maintained. 2n-2 children will be generated from the 70% of the chromosomes selected.
3. Mutation: It is selected a .01% of the population to be mutated. Of the chromosomes selected to be mutated two genes are selected to and switched in order to produce a different fitness value. These chromosomes are maintained par of the new generation. Small percentage of mutation is considered useful since using a higher percentage of mutation can affect the efficiency of the algorithm.

4.3.2 Ant Colony Optimization Model

The second optimization method used to solve the logistics system design problem is the Ant Colony Optimization algorithm (ACO).

Based on the model presented previously a model was developed to optimize the problem presented in this thesis. Each ant will represent a solution (profit) that will contain all the decision variables of the logistics system design problem. The decision variables are:

- Field of origin
- Tons of biomass
- Biomass type
- Residue destination field
- Tons of residue

A path will also be represented by each ant that will determine the operation of intermediate warehouses and type of transportation from:

- Field of origin to intermediate warehouse or biorefinery in case the option selected consists of having no intermediate warehouses open.
- Intermediate warehouse to biorefinery.
- Biorefinery to field- Biomass residue.

Each solution will be constructed by using a transition probability mass function given by

$$P_{ij} = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{l=1}^{a_i}(\tau_{il})^{\alpha}(\eta_{il})^{\beta}} & j \in AO \\ 0 \end{cases} \tag{19}$$

Where α, and β are the parameters that control the relative weight of the pheromone and the local heuristic. AO is the set of available options for the logistics system. All the decision variables previously presented will help determine the solution of each ant, which is going to be represented by the objective function,

$$\max R - \sum_{n=1}^{7} C_n \tag{20}$$

where R is our revenue and C_n are going to be the costs.

Pheromone trail intensity update is part of the ACO model presented and it will reflect the discoveries of each iteration, the intensity update is:

$$\tau_{ij}^{new} = \rho \cdot \tau_{ij}^{old} + (1-\rho) \cdot \sum_{m=1}^{E} (E - m + 1) \cdot P_m \tag{21}$$

where m = 1 will represent the best feasible solution found until now and the best E ants will be the ones that contribute their pheromone to the trail intensity where the magnitudes of contributions are weighted by their ranks in the colony. P_m represents the profit. The Pseudocode for the ACO algorithm will follow the next steps:

Step 1: Initialize:

a) Set iteration counter t=0

b) Set all parameters

c) Set pheromone trail intensities

Step 2: Determine decision variables and Construct an ant using the transition probability (equation 17).

Step 3:Evaluate the fitness of each solution (Equation 18)

Step 4: Keep the best solutions in a list. If not optimal and t < max iteration, update the pheromone trail intensity (equation 19). Step 5: Set t=t+1 and return to Step 2.

Genetic Algorithm and Ant Colony Optimization are the optimization techniques presented and described in this chapter, which were selected to approach the Biomass-to-Biorefinery problem described in Chapter 3. The next chapter will show the results obtained from the application of these two techniques to the problem.

Chapter 5: Numerical Example

The numerical analysis that is presented in this section was adapted from (Zhu and Yao 2011). A theoretical graphical layout shown in figure 23 was used to conduct the study.

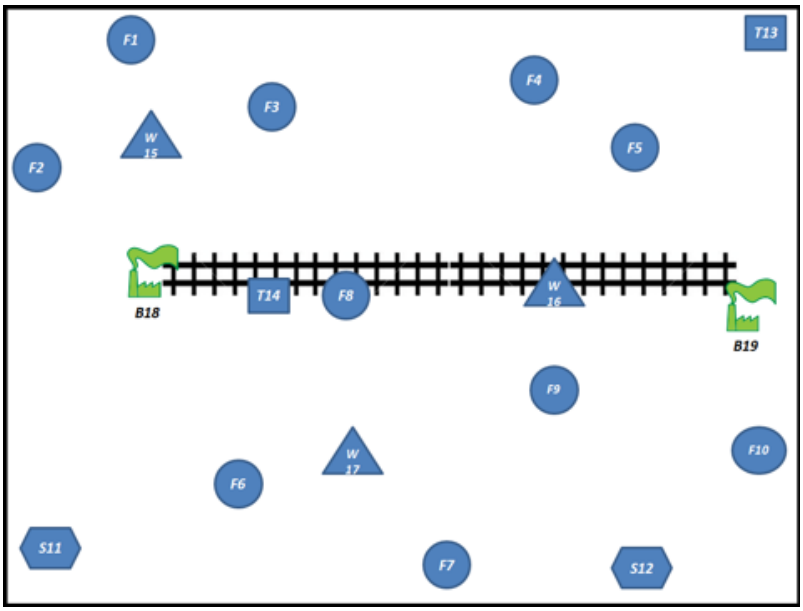


Figure 23: Geographical Layout

Geographical layout and data description:

$F_{1 \text{ to } 10}$: represent the number of switchgrass production fields

$S_{11 \text{ to } 12}$: represent the fields that produce corn stalk

$T_{13 \text{ to } 14}$: represents the fields that wheat straw

$W_{15 \text{ to } 17}$: represent the intermediate warehouses

$B_{18 \text{ to } 19}$: represent the biorefineries where bio fuel and biomass residue is produced.

Facilities accessible by train transportation: Biorefinery 18 and 19, corn stalk field 14, switchgrass field 8 and intermediate warehouse 16.

All facilities are accessible by truck.

The operating cost of each biorefinery is $v_k = \$10{,}000{,}000$ per year.

Fixed cost for intermediate warehouse (μ_j) is \$60,000 per month

Fixed cost for in-biorefinery (μ_j) is \$30,000 per month

In-field warehouse do not incur in any operating cost

Biomass residue storage ($\alpha_{ijm}, \alpha_{0jm}$) is \$2/dry ton/month in any potential warehouse.

The storage capacity ($SCAP_j$) for the in-field warehouses is unlimited, for intermediate warehouses (W_{15}, W_{16}, W_{17}) is 200,000 dry tons/month, and 60,000 dry tons/month for the in-biorefinery warehouses.

Biofuel sale price is ρ_m = \$1.8/gallon. This price was calculated according to the current potential market price.

Processing cost (C_{mkl}) for the three feedstocks is \$50/dry ton

Purchase price (λ_{m1}) for switchgrass is \$50/dry ton, and for corn stalk and wheat straw ($\lambda_{m2}, \lambda_{m3}$) is \$35/dry ton.

Conversion equivalence for the three types of biomass is 90 gallons of biofuel and .01 tons of residue from a dry ton of biomass.

Production capacity ($BCAP_{km}$) of each biorefinery is 120,000 dry tons/month.

Total yield of the three types of biomass is 2,1000,000 dry tons/year.

Harvest unit capacity ($HCAP_m$) is 7200 dry tons/month

Harvest unit annual operating and maintenance cost is γ = \$580,000.

Transportation capacity for train ($TCAP_m^2$) and truck ($TCAP_m^1$) is unlimited.

Transportation unit cost ($\beta_{ijml1}, \beta_{ijml2}$) for train is \$.04 mile/dry ton

Transportation unit cost ($\beta_{ijm01}, \beta_{ijm02}$) for truck is \$.40 mile/dry ton

Factors used in constraints are ξ =.2, δ =.5, and ψ =.005.

This model was programmed using Matlab 2012

5.1 Results

Analysis of the numerical example was performed using both meta-heuristic optimization methods (GA & ACO), both outperformed the results of the mathematical model, in the next sections description of the results of both algorithms will be presented.

5.1.1 Genetic Algorithm

The Genetic algorithm provided the best solution and it also show a better evolutionary behavior compared to ACO. The optimal profit obtained by GA was $138,483,161, and it was reached at generation 99. The total amount of gallons produced by the two biorefineries was 242,161,560 gallons, which yield revenue is $435,890,808. The unit profit is $.57/gal. Table 6 provides the costs and profit related to the best profit obtained by the GA.

Table 6: Costs, Revenue and Profit of Biofuel Production-GA

Month	1	2	3	4	5	6	7
Processing Cost	11,965,400.00	10,641,800.00	11,965,400.00	11,477,300.00	11,965,400.00	11,965,400.00	11,477,300.00
Purchasing Cost	11,965,400.00	10,641,800.00	8,375,780.00	8,034,110.00	8,375,780.00	8,375,780.00	11,477,300.00
Inventory Cost	244,786.16	244,256.72	244,786.16	244,590.92	244,786.16	244,786.16	244,590.92
Transportation Truck	96,680.43	85,134.40	96,680.43	92,736.58	957.23	96,680.43	918.18
Transportation Train	0.00	85.13	0.00	0.00	9,572.32	0.00	9,181.84
Operating Cost	1,726,666.67	1,726,666.67	1,726,666.67	1,726,666.67	1,726,666.67	1,726,666.67	1,726,666.67
Harvest Unit	1,498,333.33	1,498,333.33	1,498,333.33	1,498,333.33	1,498,333.33	1,498,333.33	1,498,333.33
Revenue	$38,767,896.00	$34,479,432.00	$38,767,896.00	$37,186,452.00	$38,767,896.00	$38,767,896.00	$ 37,186,452.00
Profit	$11,270,629.41	$ 9,641,355.75	$14,860,249.41	$14,112,714.50	$14,946,400.29	$14,860,249.41	$ 10,752,161.06
Month		8	9	10	11	12	Total
Processing Cost		11,965,400.00	6,190,800.00	11,965,400.00	11,477,300.00	11,477,300.00	134,534,200.00
Purchasing Cost		11,965,400.00	6,190,800.00	11,965,400.00	11,477,300.00	11,477,300.00	120,322,150.00
Inventory Cost		244,786.16	242,476.32	244,786.16	244,590.92	244,590.92	2,933,813.68
Transportation Truck		96,680.43	50,021.66	96,680.43	92,736.58	92,736.58	898,643.39
Transportation Train		0.00	0.00	0.00	0.00	0.00	18,839.29
Operating Cost		1,726,666.67	1,726,666.67	1,726,666.67	1,726,666.67	1,726,666.67	20,720,000.00
Harvest Unit		1,498,333.33	1,498,333.33	1,498,333.33	1,498,333.33	1,498,333.33	17,980,000.00
Revenue		$38,767,896.00	$20,058,192.00	$38,767,896.00	$37,186,452.00	$37,186,452.00	$ 435,890,808.00
Profit		$11,270,629.41	$ 4,159,094.02	$11,270,629.41	$10,669,524.50	$10,669,524.50	$ 138,483,161.63

The highest cost was processing cost as shown in figure 24, followed by purchasing cost, operating cost, harvesting units, inventory cost, transportation by truck and the lowest cost transportation by train.

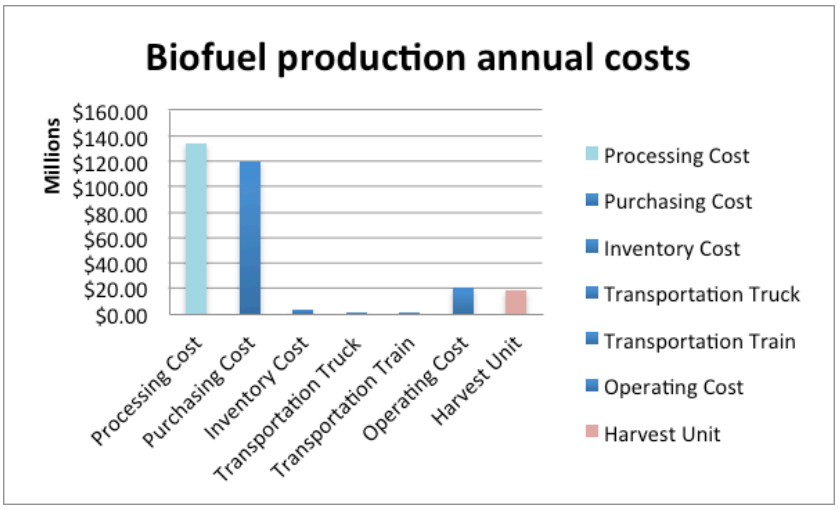


Figure 24: Biofuel Production Annual Costs

As part of the optimal design shown in table 7 and 8 it was determined by the algorithm that intermediate warehouses were closed during the year. By not operating intermediate warehouse no additional operation cost is generated. All biomass is to be stored in the in-field warehouses and in-biorefinery warehouses. Transportation of biomass is direct from field of origin to biorefinery by either train or truck. When available train transportation was selected as the transportation mode due to its low cost. The total amount produced of biomass residue by both biorefineries was 26,906 tons and was recirculated to fields 1, 2, 7 & 8. The 4th highest cost was harvesting unit, the optimal design selected a total of 31 harvesting units for the entire year. The stopping criterion of the algorithm was 100 iterations.

Table 7: GA optimal design

Design	Month 1	Month 2	Month 3	Month 4	Month 5	Month 6	Month 7	Month 8	Month 9	Month 10	Month 11	Month 12
Biomass Type	1	1	3	2	3	3	1	1	1	1	1	1
Field of Origin	7	6	12	13	14	11	10	9	1	5	2	8
Intermediate Warehouse	0	0	0	0	0	0	0	0	0	0	0	0
Transportation type to warehouse	0	0	0	0	0	0	0	0	0	0	0	0
Transportation type to biorefinery	1	1	1	1	2	1	2	1	1	1	1	1
Residue Transportation	1	2	1	1	1	1	1	1	1	1	1	1

Table 8: Biomass type and type of transportation by month

Biomass Type and Type of Transportation by Month	
Month 1: Biomass Switchgrass, Transportation is truck	**Month 7:**Biomass Switchgrass, Transportation Truck, train
Month 2: Biomass Switch Grass, Transportation truck and train	**Month 8:**Biomass Switchgrass, Transportation Truck
Month 3: Biomass Wheat Straw, Trnasportation Truck	**Month 9:**Biomass Switchgrass, Transportation Truck
Month 4: Biomass Wheat Straw, Transportation Truck	**Month 10:**Biomass Switchgrass, Transportation Truck
Month 5: Biomass Corn Stalk, Transportation truck and train	**Month 11:**Biomass Switchgrass, Transportation Truck
Month 6: Biomass Wheat Straw, Transportation truck	**Month 12:**Biomass Switchgrass, Transportation Truck

5.1.2 Ant Colony Optimization

ACO underperformed compared to the GA but still provided good results. A total of ten ants were used in the algorithm and initialization parameters considered were: τ_0=.005, α=3 and β=12. The optimal profit of the system having three different feedstocks was $100,331,744.94, reached during iteration 61. The total amount of gallons produced by the two biorefineries was 191,908,800 gallons, which yield revenue is $345,435,840. The unit profit is $.52/gal. Table 9 provides the costs and profit related to the best profit obtained by ACO.

Table 9: Costs, Revenue and Profit of Biofuel Production-ACO

Month	1	2	3	4	5	6	7
Processing Cost	11236500	10109100	8684700	7350500	5718800	11093700	8973900
Purchasing Cost	11236500	10109100	6079290	5145350	4003160	7765590	8973900
Inventory Cost	229224.6	446225.64	350861.88	296960.2	231039.52	448185.48	423067.56
Transportation Truck	90790.92	81681.528	70172.376	59392.04	45751.2	887.496	717.912
Transportation Train	0	0	0	0	0	8874.96	7179.12
Operating Cost	1726666.667	1786666.667	1726666.667	1726666.667	1726666.667	1726666.667	1786666.667
Harvest Unit	1256666.667	1256666.667	1256666.667	1256666.667	1256666.667	1256666.667	1256666.667
Revenue	$ 36,406,260.00	$ 32,753,484.00	$ 28,138,428.00	$ 23,815,620.00	$ 18,528,912.00	$ 35,943,588.00	$ 29,075,436.00
Profit	$ 10,629,911.15	$ 8,964,043.50	$ 9,970,070.41	$ 7,980,084.43	$ 5,546,827.95	$ 13,643,016.73	$ 7,653,338.07
Month		8	9	10	11	12	Total
Processing Cost		9450300	4851000	10420900	9170300	9556300	106,616,000.00
Purchasing Cost		9450300	4851000	10420900	9170300	9556300	96,761,690.00
Inventory Cost		621792.12	338960.4	452586.36	427074.12	434948.52	4,700,926.40
Transportation Truck		151960.824	39196.08	84200.872	147458.424	77214.904	849,424.58
Transportation Train		0	0	0	0	0	16,054.08
Operating Cost		1726666.667	1786666.667	1786666.667	1786666.667	1786666.667	21,080,000.00
Harvest Unit		1256666.667	1256666.667	1256666.667	1256666.667	1256666.667	15,080,000.00
Revenue		$ 30,618,972.00	$ 15,717,240.00	$ 33,763,716.00	$ 29,711,772.00	$ 30,962,412.00	$ 345,435,840.00
Profit		$ 7,961,285.72	$ 2,593,750.19	$ 9,341,795.43	$ 7,753,306.12	$ 8,294,315.24	$ 100,331,744.94

The highest cost was processing cost as shown in figure 25, followed by purchasing cost, operating cost, harvesting units, inventory cost, transportation by truck and the lowest cost transportation by train, showing a same distribution of costs as in GA.

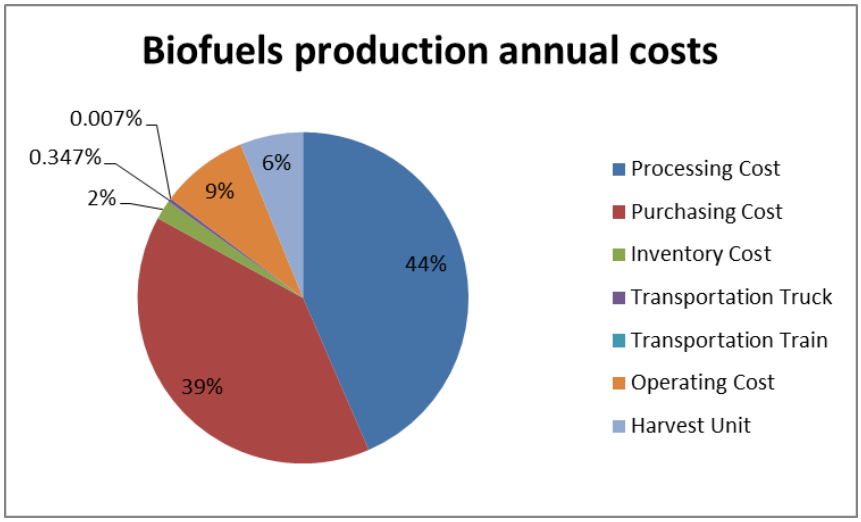


Figure 25: Biofuel Production Annual Costs-ACO

The optimal design shown in table 10 & 11 that the algorithm determined included opening intermediate warehouse 17 during the 2nd month, intermediate warehouse 16 during the 7th month, intermediate warehouse 15th during the 9th, 11th and 12th month, and intermediate warehouse 17 during month 10th. ACO had 6 intermediate warehouses opened during the twelve-month period while GA had all of them close during the year. By operating intermediate warehouse an additional operation cost was generated decreasing profit. All biomass is to be stored in the in-field warehouses and in-biorefinery warehouses when no intermediate warehouses are opened. Transportation of biomass is direct from field of origin to biorefinery by either train or truck on the following months: 1, 3, 4, 5, 6, and 8. When available train transportation was selected as the transportation mode due to its low cost. The total amount produced of biomass residue by both biorefineries was 21,323 tons and was recirculated all switchgrass fields. The 4th highest cost was harvesting unit, the optimal design selected a total of 26 harvesting units for the entire year. The stopping criterion of the algorithm was 100 iterations.

Table 10: ACO optimal design

Design	Month 1	Month 2	Month 3	Month 4	Month 5	Month 6	Month 7	Month 8	Month 9	Month 10	Month 11	Month 12
Biomass Type	1	1	2	3	2	3	1	1	1	1	1	1
Field of Origin	4	7	12	13	12	14	1	6	9	8	2	5
Intermediate Warehouse	0	17	0	0	0	0	16	0	15	17	15	15
Transportation type to warehouse	0	1	0	0	0	0	1	0	1	1	1	1
Transportation type to biorefinery	1	1	1	1	1	2	2	1	1	1	1	1
Residue Transportation	1	1	1	1	1	1	1	1	1	1	1	1

Table 11: Biomass type and transportation type by month

Biomass Type and Type of Transportation by Month	
Month 1: Biomass Switchgrass, Transportation is truck	**Month 7:**Biomass Switchgrass, Transportation Truck, train and truck
Month 2: Biomass Switch Grass, Transportation truck	**Month 8:**Biomass Switchgrass, Transportation Truck
Month 3: Biomass Corn Stalk, Trnasportation Truck	**Month 9:**Biomass Switchgrass, Transportation Truck
Month 4: Biomass Wheat Straw, Transportation Truck	**Month 10:**Biomass Switchgrass, Transportation Truck
Month 5: Biomass Corn Stalk, Transportation Truck	**Month 11:**Biomass Switchgrass, Transportation Truck
Month 6: Biomass Wheat Straw, Transportation Truck and Train	**Month 12:**Biomass Switchgrass, Transportation Truck

5.2 Sensitivity Analysis

A sensitivity analysis was performed in this section for both algorithms to know how changing the parameters can affect the outcome of a solution.

5.2.1 Genetic Algorithm

Selection percentage, mutation percentage, population size and number of iterations are the parameters that will be changed in order to determine if the outcome of the solution can be affected. The values assigned to the parameters initially are:

Iterations: 100

Population Size: 50 chromosomes

Selection percentage: 70%

Mutation percentage: 1%

Number of iterations will be varied from 125 to 400 iterations since it will give an opportunity to the algorithm to keep evolving and having a higher probability of getting a better optimal solution. The population size is going to be between 55 and 110 to see if having a bigger population size will give a bigger range of solutions. Selection was varied between 72% and 94%, having a higher rank percentage will lead to a larger variety of solutions. Mutation was varied between 1% and 3.2% to see if chromosomes can improve results or can lead to poor results since having a larger mutation percentage can lead to defects in chromosomes.

Table 12: GA Sensitivity Analysis Results

Generations	Population Size	Selection %	Mutation %	Profit	Revenue	Generation	Gallons
125	55	72	1	$ 143,033,491.00	$ 449,066,268.00	109	249,481,260.00
150	60	74	1.2	$ 138,856,104.00	$ 428,070,096.00	124	237,816,720.00
175	65	76	1.4	$ 142,800,298.00	$ 441,032,364.00	123	245,017,980.00
200	70	78	1.6	$ 147,395,825.00	$ 443,959,056.00	136	246,643,920.00
225	75	80	1.8	$ 136,336,032.00	$ 379,143,828.00	177	210,635,460.00
250	80	82	2	$ 145,350,625.00	$ 445,025,664.00	250	247,236,480.00
275	85	84	2.2	$ 142,115,993.00	$ 449,514,036.00	227	249,730,020.00
300	90	86	2.4	$ 146,427,818.00	$ 429,692,688.00	289	238,718,160.00
325	95	88	2.6	$ 145,056,110.00	$ 413,602,848.00	293	229,779,360.00
350	100	90	2.8	$ 143,729,761.00	$ 419,717,376.00	150	233,176,320.00
375	105	92	3	$ 148,461,238.00	$ 458,405,568.00	246	254,669,760.00
400	110	94	3.2	$ 145,552,545.00	$ 427,868,244.00	157	237,704,580.00

Table 12, shows the outcome profits after varying the parameters. The best profit was obtained when the parameters equal to 375 Generations, 105 Population size, 92% Selection and 3% mutation, the optimal profit was reached after 246 generations.

5.2.2 Ant Colony Optimization

ACO algorithm depends on several user-defined parameters that will control the behavior of the algorithm and will impact the outcome. A sensitivity analysis was used in order to determine how the outcome was affected by the next parameters: α, β, ρ and τ_0, initially the values assigned were 3, 6, .05 and .005 respectively. ρ was fixed at .5 which is the one controlling the pheromone evaporation rate and is considered the best value for this parameter according to suggestions in earlier studies (Bonbeu et al. 2000). Parameter α is the relative importance of the pheromone intensity, several analysis have been

performed on determining what value is optimal for this parameter and it was determined that the optimal values are between 0 and 3 (Wai, Holger and Simpson 2005). As for β which is the relative importance given to the heuristic part, it was determined that the optimal range is between 0 and 2 according to previous studies (Wai, Holger and Simpson 2005), in this analysis a wider range was analyzed to see if the outcome could be improved if it was given a higher weight to the heuristic part. Also the number of ants was ranged between 6 and 17, iterations between 100 and 375 and τ_0 between .001 and .0065 to see if the having a wider search space and giving opportunity to the algorithm to evolve could improve the algorithm.

Table 13: ACO Sensitivity Analysis Results

# of Ants	Iterations	τ0	α	β	Profit	Revenue	Gallons
6	100	0.001	1	6	$98,326,937.00	$344,800,476.00	191,555,820
7	125	0.0015	1.5	7	$101,396,247.00	$340,971,444.00	189,428,580
8	150	0.002	2	8	$106,275,300.00	$357,907,896.00	198,837,720
9	175	0.0025	2.5	9	$110,378,149.00	$372,805,740.00	207,114,300
10	200	0.003	3	10	$112,836,246.00	$371,797,128.00	206,553,960
11	225	0.0035	3.5	11	$108,567,841.00	$371,462,436.00	206,368,020
12	250	0.004	4	12	$106,820,237.00	$367,764,948.00	204,313,860
13	275	0.0045	4.5	13	$107,608,663.00	$364,468,248.00	202,482,360
14	300	0.005	5	14	$103,729,531.00	$359,035,740.00	199,464,300
15	325	0.0055	5.5	15	$103,747,004.00	$355,351,536.00	197,417,520
16	350	0.006	6	16	$94,603,233.00	$332,705,880.00	184,836,600
17	375	0.0065	6.5	17	$100,851,129.00	$346,355,028.00	192,419,460

The results shown in table 13 determined that the optimal α, β, and $\tau_0$values for this problem were 3, 10, .003 with a total of 10 ants and 200 iterations.

Chapter 6: Conclusions and Future Research

After describing the problem to be solved in chapter 1 and chapter 3, reviewing the mathematical approaches to the problem and analyzing several meta-heuristic optimization methods in chapter 2, the application of the Genetic Algorithm and Ant Colony Optimization algorithm (ACO) to solve the logistics system design of biomass-to-biorefinery problem was presented in Chapter 4, an example is presented in chapter 5 in order to show the performance of the algorithms, and finally this chapter will present the conclusions and future research.

This thesis established a single objective Genetic Algorithm and Ant Colony Optimization algorithm model for a logistics system design with multiple types of feedstock. Genetic algorithm is based in the mechanics of biological evolution and the Ant Colony Optimization algorithm is based on the foraging behavior of ants, these are bio-inspired metahueristic optimization tools. Both algorithms outperformed the mathematical model results. An important characteristic of the algorithms proposed is the ability to provide a solution in a short period of time in comparison with the mathematical or complete models which require exponential computational time. Out of the comparison of the two metaheuristics optimization tools the Genetic Algorithm was the one that provided the best results. Metaheuristics optimization tools utilized in this research proved that they can be as efficient as any other optimization tool used to solve and design biomass logistics system even that the do not guarantee that a global optimal solution but the do guarantee a near optimal solution.

By using metaheuristic approaches, this research proves that by these means it can be design an efficient biomass logistics systems that can help keep low the costs of producing advanced biofuels and investors can be attracted to this type of technologies and the demand for advanced biofuels can be satisfied.

Many other objectives in future research can be solved using the proposed model. Considering that the system has to be sustainable and generate a lower impact in the environment an additional

objective could be the minimization of Greenhouse gas emissions also the objective of accrued jobs can be added to the problem, since the system can create direct and indirect jobs.

The Genetic Algorithm provided the best solutions to the problem presented in this thesis, for future research it can be compared to other meta-heuristic methods which can be more appropriate to the logistics system design biomass-to-biorefinery problem such as Particle swarm Optimization (PSO), Firefly Algorithm (FA), Viral Systems (VS), and Bee Colony Optimization (BCO) and provide better results.

References

1.Abbas, H.A. 2001. MBO: marriage in honey bees optimization-a Haplometrosis polygynous swarming approach. *Proceedings of the Congress on Evolutionary Computation*, 207-214.

2.Act, A. 2007. *One Hundred Tenth Congress of the United States of America*; pp. 1-310.

3.Andras, K., Tamas V., Janos A. 2012. Constrained Particle Swarm Optimization of Supply Chains. *World Academy of Science, Engineering and Technology,* 67: 35-43.

4.Ashesh, Kr., Aditya, H.K., Chan, F.T.S., & Tiwari, M.K. 2009. Multi-Agent Based Petroleum Supply Chain Coordination: A Co-Evolutionary Particle Swarm Optimization Approach. *World Congress on Nature & Biological Inspired Computing,* 1349-1354.

5.Bartuska, A. 2006. *Why Biomass is important – The Role of the USDA Forest Service in Managing and Using Biomass for Energy and Other Uses*; pp. 1-16

6.Benatchba, K., Admane, L., Koudil, M. 2005. Using Bees to Solve a Data-Mining Problem Expressed as a Max-Sat One. *Mira, J., Álvarez, J.R. (eds.) IWINAC 2005. LNCS, vol. 3562, pp. 212–220. Springer, Heidelberg.*

7.Binder, T. 2002. Rigorous Integration of Semiconductor Process and Device Simulators. *Ph. D. thesis, Technical University of Vienna Faculty of Electrical Engineering and Information Technology.*

8.Bruglieri, M., and Liberti L. 2008. Optimally Running a Biomass-Based Energy Production Process. *Politecnico di Milano & Ecole Polytechnique.*

9.Busetti, F. 2003. Simulated Annealing Overview.

10.Chang, L., Zhongqiang, G., & Weihua, Z. 2012. A New Path Planning Method Based on Firefly Algorithm. *International Joint Conference on Computational Sciences and Optimization,* 775-778.

11.Chen, C.-W., & Fan, Y. 2012. Bioethanol supply chain system planning under supply and demand uncertainties. *Transportation Research*, 150-164.

12.Cortes., P., Garcia, J., Munuzuri, J., & Onieva, L. 2007. Viral Systems: A New Bio-Inspired Optimization Approach. *IEEE Transactions on Evolutionary Computation.*

13.Cortes, P., Garcia, J., Munuzuri,. & Guadix, J. 2010. A Viral System Massive Infection Algorithm. *International Journal of Bio-inspired Computation*, 71-74.

14.Cruz-Bernal, A. 2013. Meta-Heuristic Optimization Techniques and Its Applications in Robotics. *Polytechnic University of Guanajuato, Robotics Engineering Department.*

15.Cundiff, J., Dias, N, & Sherali, H. 1997. A linear programming approach for designing a herbaceous biomass delivery system. *Bioresource technology*, *59*, 47-55.

16.Dorigo, M., & Blum, C. 2005. Ant colony optimization: A survey. *Theoretical Computer Science*, *344*, 243-278.

17.Drias, H., Sadeg, S., Yahi, S. 2005. Cooperative Bees Swarm for Solving the Maximum Weighted Satisfiability Problem. *Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) IWANN 2005. LNCS, vol. 3512, pp. 318–325. Springer, Heidelberg*

18.Fotakis, D. G., Sidiropoul, E., Myronidis, D., & Ioannou, K. 2012. Spatial genetic algorithm for multi-objective forest planning. *Forest Policy and Economics*, 12-19.

19.Hassan R., Cohanim B., de Weck O.L., & Venter G., 2005. A Comparison of Particle Swarm Optimization and the Genetic Algorithm. *1st AIAA Multidisciplinary Design Optimization Specialist Conference, Austin, Texas*.

20.Holland, J. 1992. Genetic Algorithms: Computer Programs That "Evolve" in Ways That Resemble Natural Selection Can Solve Complex Problems Even Their Creators Do Not Fully Understand. *Scientific American*.

21.Hopp, W.J., Spearman, M.L. 2001. *Factory physics: foundations of manufacturing management* (2nd ed.). Boston: Irwin/McGraw-Hill.

22.Huang, L., & Liu, Z. 2010. A Mixed Discrete Particle Swarm Optimization for TSP. *3rd International Conference on Advanced Computer Theory and Engineering*, 2, V2-208-V2-211.

23.Ituarte-Villareal, C., & Espiritu, J. 2011. Optimization of Wind Turbine Placement Using Viral Based Optimization Algorithm. *Elsevier B.V*.

24.Jayaraman, V., & Ross, A. 2008. An evaluation of new heuristics for the location of cross-docks distribution centers in supply chain network design. *Computers & Industrial Engineering, 55,* 64-79.

25.Judd, J., Sarin, S., Cundidd, J.S., & Grisso, R.D. 2010. An Optimal Storage and Transportation System for a Background / Motivation. In *2010 ASABE Annual International Meeting*, 0300, 1-15.

26.Jung, W., & Young, H. 2010. Heuristic algorithms for production and transportation planning through synchronization of a serial supply chain. *International Journal of Production Economics*, 123, 433-447.

27.Karaboga, D. 2005. An idea based on honey bee swarm for numerical optimization (Technical Report-Tr06, October, 2005), *Erciyes University, Engineering Faculty Computer Engineering Department Kayseri/Türkiye*.

28.Kennedy, J., & Eberhart, R. 1995. Particle Swarm Optimization. *IEEE International Conference on Neural Networks*, 4, 1942-1948.

29.Kirkpatrick, S., Gelatt, C.D., & Vecchi, M.P. 1983. Optimization by Simulated Annealing. *Science*, 220, 671-680.

30.Lee, K.S., & Geem, Z.W. 2005. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer methods in applied mechanics and engineering*, 194(36), 3902-3933.

31.Lourenco, H. 2005. Supply Chain Management: An opportunity for Metaheruistics. *Operations Research/Computer Science Interface Series*, 30, 329-356.

32.Lucic, P., & Teodorovic, D. 2001. Bee System: Modeling Combinatorial Optimization Transportation Engineering Problems by Swarm Intelligence. In *The Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis,* 441.445.

33.Lucic, P., & Teodorovic, D. 2002. Transportation modeling: and artificial life approach. *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence*, 215-223.

34.Lucic, P., & Teodorovic, D. 2003. Vehicle routing with uncertain demand at nodes: the bee system and fuzzy logic approach. *Verdegay, J.L. (ed.) Fuzzy Sets in Optimization*, 67-82.

35.Lucic, P., & Teodorovic, D. 2003. Computing with Bees: Attacking Complex Transportation Engineering Problems. *International Journal on Artificial Intelligence Tools,* 12, 375-394.

36.Mapemba, L., Epplin, F., Taliaferro, C., & Huhnke, R. 2004. Use of Conservation Reserve Program Land for Biorefinery Feedstock Production. In *Proceedings from the American Agricultural Economics Association annual meetings*, 1-35.

37.Marvin, W.A., Schmidt, L.D., Benjaafar, S., Tiffany, D.G., & Daoutidis, P. 2012. Economic Optimization of a Lignocellulosic Biomass-to-Ethanol Supply Chain. *Chemical Engineering Science*, 68-79.

38.Mayntz, R. 2003. Mechanisms in the Analysis of Macro-Social Phenomena. *MPI Working paper; re-printed in Philosophy of the Social Sciences*, 34(2), 237-259.

39.Metropolis, N., Rosenbluth, A., Rosenbluth, M., & Teller, A.H. 1953. Equation of State Calculations by Fast Computing Machines. *The Journal of Physics* 21, no.6.

40.Mitchell, R.B., Vogel, K.P., Schemer, M.R., & Pennington, D. 2010. Switchgrass for Biofuel Production. Sustainable AG. Energy Community of Practice, Extension, Available from: *http://extension.org/pages/Switchgrass_for_biofuel_production/*.

41.Mohd, N., & Geraghty, J. 2011. Genetic Algorithm Performance with Different Selection Strategies in Solving TSP. *International Conference of Computational Intelligence and Intelligent Systems*.

42.Noor, S., Khan, M.K., Hussain, I., & Ullah, I. 2007. Operational Scheduling of traditional and flexible manufacturing systems using genetic algorithms, artificial neural networks and simulation. *PhD Thesis, University of Bradford,* UK.

43.Obtiko, M. 1998. Introduction to genetic algorithms. *Czech Technical University*. URL: http://cs.felk.cvut.cz/* xobitko/ga.

44.Purnomo, H. D., Wee, H., Praharsi, Y. 2012. Two inventory review policies on supply chain configuration problem. *Computers and Industrial Engineering*. Volume 63 Issue 2, 448-455

45.Qing, W., Han-Chao, Z. 2011. Optimization of task allocation and knowledge workers scheduling based-on particle swarm optimization. *International Conference on Electric Information and Control Engineering*, 574-578.

46.Sato, T., Hagiwara, M. 1997. Bee System: Finding Solution by a Concentrated Search. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics Computational Cybernetics and Simulation*, 3954-3959.

47.Shastri, Y., Hansen, A.C., Rodriguez, L.F., & Ting, K.C. 2011. Development and application of BioFeed model for optimization of herbaceous biomass feedstock production. *Biomass Bioenergy*, 35(7), 2961-2974.

48.Stowe, N. 2012. Renewable Biomass Can Help Reduce U.S. Petroleum Dependence: Farm Bill Energy Programs Are Key. *Environmental and Energy Study Institute*.

49.Subbu, K., & Satish, J. 2012. Optimal biomass-harvesting model for Biobuthanol biorefineries. pp. 4-5.

50.Sulaiman, M.H., Mustafa, A., Ailman O., and Rahim, S.R. 2012. Optimal Allocation and Sizing of Distributed Generation in Distribution System via Firefly Algorithm. *IEEE International Power Engineering and Optimization Conference*, 84-89.

51.Sulaiman, M.H., Mustafa, M.W., Zakaria, N.Z., Aliman, O., & Rahim, R. 2012. Firefly Algorithm Technique for Solving Economic Dispatch Problem. *IEEE International Power Engineering and Optimization Conference*, 90-95.

52.Suriyadi, D., & Kusnadi, E.K. 2011. Viral Systems Application for Knapsack Problem. *Third Computational Intelligence, Communication Systems and Networks*. 11-16.

53.Tembo, G., Epplin, F., & Huhnke, R. 2003. Integrative investment appraisal of lignocellulosic biomass-to-ethanol industry. *Journal of Agricultural and Resource Economics*, 611-633.

54.Teodorovic, D., Dell'Orco, M. 2005. Bee Colony Optimization – a cooperative learning approach to complex transportation problems. *Advanced OR and AI Methods in Transportation. Proceedings of the 10th Meeting of the EURO Working Group on Transportation*, 51-60.

55.U.S. Energy Information Administration. 2012. *Monthly Energy Review*; pp.1-211.

56.Wedde, H.F., Farooq, M., Zhang, Y. 2004. BeeHive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior. LNCS, vol.3172, 83-94.

57.Yang, X. 2005. Engineering Optimizations via Nature-Inspired Virtual Bee Algorithms. *Mira, J., Álvarez, J.R. (eds.) IWINAC 2005. LNCS, vol. 3562, pp. 317–323. Springer, Heidelberg.*

58.Yang, X. 2009. Firefly Algorithms for Multimodal Optimization. *Springer-Verlag*, 169-178.

59.Yang, X. 2010. Firefly Algorithm, Stochastic Test Functions and Design Optimization.

60.Yang, X., & Deb, S. 2009. Cuckoo Search Via L'evy Flights. *Elsevier B.V.*

61.Yang, X.S. 2009. Harmony Search as a Metaheuristic algorithm. *Music-Inspired harmony Search algorithms: Theory and Applications (Editor Z.W. Geem), Studies in Computational Intelligence, Spriger Berlin*, vol. 191, pp. 1-14

62.Yonezawa, Y., Kikuchi, T 1996. Ecological algorithm for optimal ordering used by collective Honey bee behavior. *Proceedings of the Seventh International Symposium on Micro Machine and Human Science*, 249-255.

63.You, F., Tao, L., Granziano, J.D., & Snyder, S.W. 2011. Optimal Design of Sustainable Cellulosic Biofuel Supply Chains: Multi-objective Optimization Coupled with Life Cycle Assessment and Input – Output Analysis. *American Institute of Chemical Engineers*, 1157-1180.

64.Zhu, X., Li, X., Yao, Q., & Chen, Y. 2011. Challenges and models in supporting logistics system designs for dedicated-biomass-based bioenergy industry. *Bioresource technology*, *102*, 1344-51.

65.Zhu, X., & Yao, Q. 2011. Logistics system design for biomass-to-bioenergy industry with multiple types of feedstocks. *Bioresource technology, 102*, 10936-45.

Appendix A: Matlab Code Genetic Algorithm

```
tic
clear
clc
popsize=50;
generations=100;
perc=.7;
mutation=.01;
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 2 1 2 2 2
                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 2 2 2
                1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1
                1 1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 2 2 2
                1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1
                1 1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 0 0 0
                1 1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 0 0 0
                1 1 1 1 1 1 1 2 1 1 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_short(1,i)==10||months_short(1,i)==11||months_sho
rt(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);
    tttt=1;
%     m3=10.+randperm(2)
%     m4=12.+randperm(2)
    while j<=s
        if btype(1,j)==1
            field(1,j)=m2(tttt);
            j=1+size(field,2);
            tttt=tttt+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(:,:)=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_recir(1,p)=1+round(9*rand(1,1));
    end
    %%in-field warehouse
    infwareh(:,1:s)=field_recir(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;
    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(:,:);
    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
    if size(population,1)<=popsize
        x=x+1;
    else
        x=x;
    end
    end

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
        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 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

truck_transport(ll,1)=.4*(sum(transported1)*2+sum(transported2)*2+sum(transported3)
);
end
%%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
    total_switchgrass=sum(switchgrass1)*2;
    number_units(qq,1)=total_switchgrass/8/7200;
    operation_cost(qq,1)=580000*ceil(number_units(qq,1));
end
%%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;
        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);child
ren(1,49:60);children(1,61:72);children(1,73:84);children(1,85:96);children(1,97:10
8);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);chi
ldren(1,205:216);children(1,217:228);children(1,229:240);children(1,241:252);childr
en(1,253:264);children(1,265:276)];
children_red(1:popsize,:)=children(1:popsize,:);
%%mutation 1%

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,f
ff));
        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_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
clearvars child2_8 child2_9 children chrom1 chrom2 chrom3 counter dd ddd ee eee
feed_purch_cost fff gg 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
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 suma1 switchgrass1
clearvars total_processed total_switchgrass train_transport transported1
transported2 transported3 transported4 transported5 transported6 truck_transport tt
uu ww x 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
        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
                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)=.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)=.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);
        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/8/7200;
        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
        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
    clearvars children_red

child=[children(1,1:12);children(1,13:24);children(1,25:36);children(1,37:48);child
ren(1,49:60);children(1,61:72);children(1,73:84);children(1,85:96);children(1,97:10
8);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);chi
ldren(1,205:216);children(1,217:228);children(1,229:240);children(1,241:252);childr
en(1,253:264);children(1,265:276)];
    children_red(1:popsize,:)=children(1:popsize,:);
    %%mutation 1%
    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,f
ff));
            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||ch
ildren_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
    clearvars child1_8 child1_9 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
    clearvars 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');
toc
```

Appendix B: Matlab Code Ant Colony Optimization algorithm

```
clear
clc
prob=0.3;
Q=78;
t0=0.005;
alpha=3;
beta=12;
months=1:12;
numberants=10;
iterations=100;
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 2 1 2 2 2
                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 2 2 2
                1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1
                1 1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 2 2 2
                1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1
                1 1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 0 0 0
                1 1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 0 0 0
                1 1 1 1 1 1 1 2 1 1 0 0 0 0 0 0 0 0 0 0];


for ants=1:numberants
    c1=ones(4,1)
    while c1(:,:)<2
        tons=randi([30000,120000],4,12)
        sum_tons=sum(tons,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
    c=size(months,2)
    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)
%     for f=1:c
%         if btype(1,f)==1
%             switchgrass(1,f)=tons(1,f)
%             stalk_straw(1,f)=0
%         else
%             switchgrass(1,f)=0
%             stalk_straw(1,f)=tons(1,f)
%         end
%     end
    intermediate_warehouse=[0 15 16 17]
    biorefinery=[20 20 20 20]
    for h=1:c
        for g=1:4
            if g==1
                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

    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(k,j)=2*(120000+tons_resi(k,j))
                    else
                        inventory_cost(k,j)=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
                    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(k,j)-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(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(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(k,j)=2*(120000+tons_resi(k,j)+(tons(k,j)*2))
                    else

inventory_cost(k,j)=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
                            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
                    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))
                                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(k,j)-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(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(1,k);profit(1,k)]
                end
                l=l+1
            end
            desirability=profit
            probability(1,1:4)=(desirability.^beta)/sum(desirability.^beta)
            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(ants,j)=column_best
            best_option_costs(:,j)=cost_matrix(:,column_best)

        else

            for k=1:4
                if k==1
                    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(k,j)=2*(120000+tons_resi(k,j))+best_option_costs(5,j-1)
                    else

inventory_cost(k,j)=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(k,j)-truck_trans(1,k)-train_trans(1,k)-
wb_operation_cost(1,k)-harvest_unitcost(k,j)%+best_option_costs(9,j-1)

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_
unitcost(k,j);inventory_cost(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(1,k);profit(1,k)]
                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(k,j)=2*(120000+tons_resi(k,j)+tons(k,j)*2)+best_option_costs(5,j-1)
                    else

inventory_cost(k,j)=(2*((2*tons(k,j))+tons_resi(k,j)+(tons(k,j)*2)))+best_option_co
sts(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

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(k,j)-truck_trans(1,k)-train_trans(1,k)-
wb_operation_cost(1,k)-harvest_unitcost(k,j)%+best_option_costs(9,j-1)

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_
unitcost(k,j);inventory_cost(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(1,k);profit(1,k)]
                end
                l=l+1
            end
            desirability=profit
            probability(1,1:4)=(desirability.^beta)/sum(desirability.^beta)
            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(ants,j)=column_best
            best_option_costs(:,j)=cost_matrix(:,column_best)
        end
    end
    profit_ants(ants,1)=best_option_costs(9,12)
end
paths_profit=[path profit_ants]
sort_profit=sortrows(paths_profit,-13)
E=ceil(.5*numberants)
for p=1:E
    sum_E(1,p)=E-p+1
end
for o=1:numberants
    if o<=E
        pheromone(o,1)=(1-prob)*sum(sum_E)*sort_profit(o,13)
    else
        pheromone(o,1)=0
    end
end

clearvars -except prob Q t0 alpha beta months numberants iterations transportation
paths_profit E
tons_final=[];
tons_resi_all=[];
paths_profit_all=[];
best_option_cost_all=[];
forigin_all=[];
transportation_month_all=[];
for q=1:iterations
    for ants=1:numberants
        c1=ones(4,1)
        while c1(:,:)<2
            tons=randi([30000,120000],1,12)
            tons=[tons;tons;tons;tons]
            sum_tons=sum(tons,2)
            for r1=1:4
                if sum_tons(r1,1)>2100000
                    c1(r1,1)=1
                else
                    c1(r1,1)=2
                end
            end
        end
        c=size(months,2)
        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
                    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

        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(k,j)=2*(120000+tons_resi(k,j))
                        else
                            inventory_cost(k,j)=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
                        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(k,j)-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(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(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(k,j)=2*(120000+tons_resi(k,j)+tons(k,j)*2)
                        else

inventory_cost(k,j)=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
                                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
                        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))
                                    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(k,j)-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(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(1,k);profit(1,k)]
                    end
                    l=l+1
                end
                desirability=profit
                probability(1,1:4)=(desirability.^beta)/sum(desirability.^beta)
                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(ants,j)=column_best
                best_option_costs(:,j)=cost_matrix(:,column_best)
            else

                for k=1:4
                    if k==1
                        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(k,j)=2*(120000+tons_resi(k,j))+best_option_costs(5,j-1)
                        else

inventory_cost(k,j)=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(k,j)-truck_trans(1,k)-train_trans(1,k)-
wb_operation_cost(1,k)-harvest_unitcost(k,j)%+best_option_costs(9,j-1)

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_
unitcost(k,j);inventory_cost(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(1,k);profit(1,k)]
                    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(k,j)=2*(120000+tons_resi(k,j)+tons(k,j)*2)+best_option_costs(5,j-1)
                        else

inventory_cost(k,j)=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

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(k,j)-truck_trans(1,k)-train_trans(1,k)-
wb_operation_cost(1,k)-harvest_unitcost(k,j)    %+best_option_costs(9,j-1)

cost_matrix(1:9,k)=[revenue(k,j);processing_cost(k,j);purchasing_cost(k,j);harvest_
unitcost(k,j);inventory_cost(k,j);truck_trans(1,k);train_trans(1,k);wb_operation_co
st(1,k);profit(1,k)]
                    end
                    l=l+1
                end
                desirability=profit

                if j==2
                    [row1,col1]=find(path(ants,j-1)==paths_profit(1:E,j-1))
                    is1=isempty(row1)
                    if is1==1

probability(1,1:4)=(desirability.^beta)/sum(desirability.^beta)
```

```
                    else
                        for u=1:4
                            for v=1:size(row1,1)
                                if paths_profit(row1(v,1),j)==u
                                    tau(1,u)=paths_profit(row1(v,1),13)
                                    break
                                else
                                    tau(1,u)=0
                                end
                            end
                        end
                        for w=1:4

sum_profdes(1,w)=(tau(1,w).^alpha)*(desirability(1,w).^beta)
                        end
                        for x=1:4
                            if tau(1,x)==0

probability(1,x)=(desirability(1,x).^beta)/sum(desirability.^beta)
                            else

probability(1,x)=((tau(1,x).^alpha)*(desirability(1,x).^beta))/sum(sum_profdes)
                            end
                        end
                    end
                else
                    [row2,col2]=find(path(ants,j-1)==paths_profit(row1(:,:),j-1))
                    is1=isempty(row1)
                    is2=isempty(row2)
                    if is1==1||is2==1

probability(1,1:4)=(desirability.^beta)/sum(desirability.^beta)
                    else
                        for u=1:4
                            for v=1:size(row2,1)
                                if paths_profit(row1(row2(v,1),1),j)==u
                                    tau(1,u)=paths_profit(row1(row2(v,1),1),13)
                                    break
                                else
                                    tau(1,u)=0
                                end
                            end
                        end
                        for w=1:4

sum_profdes(1,w)=(tau(1,w).^alpha)*(desirability(1,w).^beta)
                        end
                        for x=1:4
                            if tau(1,x)==0

probability(1,x)=(desirability(1,x).^beta)/sum(desirability.^beta)
                            else

probability(1,x)=((tau(1,x).^alpha)*(desirability(1,x).^beta))/sum(sum_profdes)
                            end
                        end
                    end
```

```
                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(ants,j)=column_best
                best_option_costs(:,j)=cost_matrix(:,column_best)
            end
        end
        for kk=1:c
            sum_switchgrass(1,j)=switchgrass(path(ants,kk),kk)
        end
        profit_ants(ants,1)=best_option_costs(9,12)
    end
    paths_profit=[path profit_ants]
    sort_profit=sortrows(paths_profit,-13)
    E=ceil(.5*numberants)
    for p=1:E
        sum_E(1,p)=E-p+1
    end
    for o=1:numberants
        if o<=E
            pheromone(o,1)=(1-prob)*sum(sum_E)*sort_profit(o,13)
        else
            pheromone(o,1)=0
        end
    end
    best_profit_iteration(q,1)=sort_profit(1,13)
    plot(best_profit_iteration)
    tons_final=[tons_final;tons];
    tons_resi_all=[tons_resi_all;tons_resi];
    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 switchgrass cost_matrix transportation_month_all forigin_all
best_option_cost_all tons_final paths_profit_all tons_resi_all prob Q t0 alpha beta
months numberants iterations transportation paths_profit E best_profit_iteration
plot
end
title('Evolution with each Generation');
xlabel('Generation');
ylabel('Profit');
```

Vita

Isidoro Trueba was born in Ciudad Juarez, Chihuahua, Mexico, the first of two offspring of Isidoro Trueba Espinosa and Linda Dolores Rascon. He received a bachelor's degree in Industrial Engineering from the University of Texas at El Paso in Spring 2009. After graduation he started his professional career as a Quality Engineer in a packaging manufacturing company called PetroPac in Ciudad Juarez, Chihuahua. Then in fall 2011 he started the Master's in Industrial Engineering while working with Dr. Heidi Taboada as a Research Assistant. He had the opportunity to present his research about Feedstock Logistics in the 2013 ISERC Annual Conference held in San Juan, Puerto Rico.

Permanent address: 3641 Almond Beach Dr.
El Paso, TX, 79936

This thesis/dissertation was typed by Isidoro Trueba.