Systems & Management 10 (2015), pp 561-574

SYSTEMS & MANAGEMENT
Electronic magazine

MIXED INTEGER LINEAR PROGRAMMING AND GENETIC ALGORITHM APPLIED TO STORAGE AND TRANSPORTATION PROBLEMS IN AN OIL INDUSTRY

Angela Olandoski Barboza[a], Flavio Neves Junior[a], Silvana Ligia Vincenzi Bortolotti[a], Rosely Antunes de Souza[a]

aFederal Technologic University of Parana (UTFP)

Abstract

The advancement of the international services and products market and the constant exchange of information challenge managers to seek new paths for their companies. Following this, new technologies are also desired, as they help achieving a better operational efficiency. The Brazilian oil industry, particularly, has been investing in applied researches, development, and technological capacitation to remain competitive in the international market. This productive segment comprehends many issues that must be further studied. These include problems such as products storage and transportation. This research approaches a production scheduling involving diesel storage and distribution in an oil refinery. To find the solution, a Mixed Integer Linear Programming (MILP) was first used, in a discrete-time system. The model was solved using LINGO 8.0 computer software. Next, a methodology was developed applying Hybrid Steady State Genetic Algorithm integrated to Linear Programming for solving the same model. After conducting tests comparing the MILP model with the new methodology and analyzing the results obtained, it was concluded that the new method resulted in a higher quality of problems solution and computational time.

Keywords: Scheduling, Mixed Integer Linear Programming, Genetic Algorithm.

1. INTRODUCTION

Due to technological developments, globalization, important companies mergers and to environmental awareness, global marketing has been changing. Economic changes, market dynamics and increasing competition are all part of world globalization, which in turn, intensifies international trade of services and products and promotes cultural exchange and a recurring share of information. Such modifications may force organizations to create innovate solutions to stay in the market, as they signify raising the competition. Facing this scenario, the companies now seek a new management model based mainly on reducing costs and profit margins of their products. Also, they strive considerable improving distribution related services so they can compete with other companies. Generally speaking, production costs and products quality tend to match regardless of the company responsible for the manufacturing process. For that reason, optimizing operations - that is, making products reach their final customer with a fair cost, in the right amount and within expected time - is the greater differential.

In the context of the industries, the major problem is optimizing the product system, in which production management and planning are of great importance. Optimizing is essential to keep the competitivity between industries and operating costs reduction flowing. It involves establishing production rates, material maintenance policy, adjusting strategies, assigning and designating products to their respective machines, and, finally, regulating a policy for delivering the goods to the customer.

Nowadays, brazilian companies are going through a phase in which capital gains means survival. Just in time manufacturing (Lean Manufacturing), i.e. an organizational structure prepared to meet product demand at any time using reduced supplies, offers advantages related to productivity, efficiency and quality. Given that, it becomes useful devel-

PROPPI / LATEC
DOI: 10.20985/1980-5160.2015.v10n4.448

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

oping techniques that can assure entrepreneurs and managers they are working with a product system that, when well managed, can increase competitive advantages.

Oil refineries are one of many productive sectors that seek performance improvement. One of the problems that need optimizing in this sector is production planning (scheduling). Part of this problem involves products storage and distribution. Due to its combinatorial feature, the optimal solution is hard to reach. For this reason, has been the subject of many researches, such as Jie *et* Ierapetritou (2003), Joly Moro *et* Pinto (2002) and Wu *et al.* (2005). Moro (2000) emphasizes that many issues involving oil refinery are described as problems of Mixed Integer Linear Programming (MILP) and therefore solved with the help of commercial computational packages. Therefore, due to its nature, the increasing number of integer variables makes the utilization of these software impracticable, as they demand excessive computational time.

This background justifies the pursuit for other approaches to solve this problem. In consequence, this research was done focusing in finding a more efficient computation methodology compared to classic ones (the Branch-and-Bound method, for instance) in order to solve a MILP problem. It specifically sought developing a MILP model to solve a problem related to diesel transportation and storage in an oil refinery, expressing a discrete time and decide applying the Branch-and-Bound method. Then, a new methodology was developed to solve the same model, applying Genetic Algorithm integrated to Linear Programming (PL). The choice to apply Genetic Algorithm when searching for adequate solutions to solve the problem here discussed within a reasonable computational time was made due to its validated efficiency, versatility and strength. Among the existing algorithms, some that could also be used are: Tabu Search, Simulated Annealing and Differential Evolution to solve discrete problems.

MILP and Genetic Algorithm have been used to solve many optimization problems not only in oil refineries, but also in other fields. Some researches as Thunberg *et* Ögren (2011), Subbaraj, Rengaraj *et* Salivahanan (2011), Wang *et* Tang (2011), Zhang *et* Shang (2009), Floudas *et* Lin (2005), Duan *et* Wang (2011), Morabito *et* Abuabara (2008), Carrano *et al.* (2005), Almeida *et al.* (2003) and Liu *et al.* (2011) was developed using these operational research techniques.

2. 2. THEORETICAL FRAMEWORK

1.1 Linear Programming (LP)

Linear Programming (LP) is one of the most applied and important methods of Operational Research. Both the simplicity of the model involved and the availability of an adequate solution technique programmable in computers, such as the Simplex method - described by Dantzig (1963) - makes applying LP easy. This technique is widely used, as it has the ability to model important and complex decision problems. The Simplex method description can be found on Zionts (1974). An LP problem consists of:

1. A linear function formed with decision variables called objective function, whose value must be optimized;

2. A interdependent relationship between decision variables, called constraints, expressed by a system of linear equations or inequations, called constraints;

3. Decision variables that must either be positive or null.

Equation (1) demonstrates LP problem formulation:

$$\begin{aligned} &\text{maximize (ou minimize)} \\ &z = \sum_{j\in N} c_j x_j,\ N = \{1,\ldots,n\} \\ &\textit{sujeito a} \\ &\sum_{j\in N} a_{ij} x_j\ (\leq, = ou \geq)\, b_i,\ \mathrm{i} \in \mathrm{M} = \{1,2,\ldots,\mathrm{m}\} \\ &\mathrm{x}_j \geq 0,\ \mathrm{j} \in \mathrm{N} \end{aligned} \tag{1}$$

where c_j, a_{ij} and b_i are known constants to any i and j; x_j are non-negative variables. Problem contraints can be molded into equations by adding a (non-negative) slack variable x_{n+i}, if *i-th* is ≤, and by subtracting a (non-negative) slack variable, x_{n+k}, if *k-th* is ≥. Assuming that when slack variables are introduced, m + n variables appears. Then, it can be represented in matrix form, as shown in equation (2):

$$\begin{aligned} &\textit{maximize (ou minimize)} \\ &\quad z = cx \\ &\textit{sujeito a} \\ &\quad Ax \ = b \\ &\quad x \geq 0 \end{aligned} \tag{2}$$

where *c* is a vector line of *(n + m)* order, *A* is a matrix *m' (m+n)*, *x* is a vector column of *(m + n)* order and *b* is a vector column of order *m*.

2.1.1 Integer Programming (IP) and Mixed Integer Programming (MIP)

Real life problems demand using variables that can assume only integer values. This characterizes an Integer Programming (IP) problem. Equation (3) demonstrates this type of problem:

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

$$\begin{aligned}
&\text{maximize (ou minimize)}\\
&\quad z = g_0(x_1, x_2, \ldots, x_n)\\
&\text{sujeito a}\\
&\quad g_i(x_1, x_2, \ldots, x_n)(\leq ou = ou \geq) b_i,\, i \in M = \{1,2,\ldots,m\}\\
&\quad x_j \geq 0,\, j \in N = \{1,2,\ldots,n\}\\
&\quad x_j \text{ inteira, } j \in I \subseteq N
\end{aligned} \tag{3}$$

where $x_j, j \in N$ are the variables, $g_i, i \in M \cup \{0\}$ are variable functions $x_1, x_2, \ldots, x_n$, and $b_i, i \in M$ are known constants. If $I = N$, i.e. all variables are integer, then the problem above is necessarily a IP one. If $I \subset N$, it is said to be a Mixed Integer Programming (MIP) problem.

2.1.2 Integer Linear Programming (ILP) and Mixed Integer Linear Programming (MILP)

In most IP problems, functions $g_i, i \in \{0\} \cup M$ in equation (3) are linear and the model can be described as shown in equation 4:

$$\begin{aligned}
&\text{maximize (ou minimize)}\\
&\quad z = \sum_{j \in N} c_j x_j,\, N = \{1,\ldots,n\}\\
&\text{sujeito a}\\
&\quad \sum_{j \in N} a_{ij} x_j (\leq, \geq ou =) b_i,\, i \in M = \{1,\ldots,m\}\\
&\quad x_j \geq 0,\, j \in N\\
&\quad x_j \text{ inteira, } j \in I \subseteq N
\end{aligned} \tag{4}$$

where c_j, a_{ij} and b_i are known constants to any *i* and *j*, and x_j are non-negative variables. Which states that if $I = N$, i.e. all variables are integer, then we have an Integer Linear Programming problem. If $I \subset N$, it is said to be a Mixed Integer Linear Programming (PLIM) problem.

Many practical ILP models restrict integer variables to only take values of "0" or "1"; these are called Binary Integer Linear Programming problems. These variables are used to make yes ("1") or no ("0") decisions.

Many scheduling problems can be expressed as MILP problems as their mathematical optimization models involve continuous and discrete variables that must satisfy a set of inequality and equality linear constraints (Moro, 2000). Resolution for Mixed Integer Linear Optimization problems, understood as to obtain an optimal solution, can be harsh because of their combinatorial nature. It is considered that the space of integer solutions consists of a finite number of points. Even on mixed cases, the search space is primarily defined by integer variables. An enumerated method is a data method that analyzes every point, even when using its most basic language. This can be defined as an exhaustive search. This simple method can become even more efficient if enumerate only part of the candidate's solutions while discarding the non-promising points. The efficiency of a search algorithm relies on its capacity to eliminate non promising solution points. The Branch-and-Bound method (Zionts, 1974) and implicit enumeration (Taha, 1975). These techniques allow using problem relaxation strategies for estimating within reasonable time the value of the best solution that can be found in each segment of the enumeration.

2.2 Computational Tools for Optimization Problems

Many multifaceted software are available for solving research and optimization problems. Developments seen in LP motivated breakthroughs to solve these problems. Linear Programming solving software use the Simplex method and/or the interior point search method. For ILP and MILP problems, most software use the Branch and Bound method.

Pinto (2000) made a chart listing several software and its details, such as: developer information, computing platform used, input formats, problems that can be solved through the software and farther relevant data.

Among these software, it is important to point out:

LINGO 8.0 (LINDO Systems Inc., 2002): of easy application, this solver is able to analyze and find optimal solutions for both large Linear and Non-Linear Programming problems. Integer models are solved using the branch-and-bound method.

ILOG CPLEX 8.0 (ILOG CPLEX, 2002): developed to solve LP problems, the software also solves Quadratic Programming, MIP and Network Flow problems.

2.3 Genetic Algorithm (GA)

Initially proposed by Holland (1975), GA was inspired by the biological mechanism of evolution, which was based on Darwin's works about the origin of the species and natural genetic – the last being primarily due to Mendel. Among GA definitions found in literature, it is worth mentioning Tanomaru's (1995), who defines it as a computational research method based on natural evolution and genetics mechanisms. In a GA, a population of candidate solutions to the problem evolves accordingly to probabilistic operators conceived through biologic metaphors; generations tend to present better solutions each time the evolving process happens. GA is considered efficient to generate optimal or near-optimal solutions to a variety of problems, as it doesn't present many ~~the~~ of the limitations found in traditional research methods. Besides, in most cases, GA is capable to find the solution to problems that other optimization strategies were unable to conclude.

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

GA differs from conventional optimization and search procedures in several fundamental ways. Goldberg (1989) summarized it in four aspects:

- works with a population of points instead of a single point;
- works in a space of solution coding, not in the space of direct search;
- requires only information about value of objective function for each population member, and do not require derivative or other auxiliary knowledge;
- uses probabilistic transition, not deterministic rules.

GA works with an array of individuals (population); each individual represents a feasible solution. The function to be optimized represents the environment, in which the initial population is located. Due to species evolution and natural genetic mechanisms, it is expected that the fittest individuals will be more likely to reproduce. It is also hoped that every new generation will be best fitted to the environment. The next generation will an evolution of the last one and for this to happen the fittest individuals should have higher probability of been selected to compose the new breed. However, random samples of the population can also be used. Selection is the stage during which individuals are selectively chosen. The next step is applying genetic operators, which will act on genotypes generating new individuals, also called search mechanisms (Holland, 1975). Among those mechanisms, the most frequently applied are crossover (also called recombination) and mutation. If selection, recombination and mutation operations succeed, a better generation will be hopefully created.

3. PROBLEM MODEL

This work developed a technique to solve a short term scheduling problem in a sub-system of an oil refinery. The complexity of production planning operations, which tend to create large combinatorial problems, and computational processing limitations justifies the search for techniques that are more efficient. Scheduling problems in oil refineries are formulated as mixed integer programming models and are solved using exact techniques; however, implementing these methods to find the solution is impracticable as they can be very time consuming. The subsystem studied in this research involves activities such as diesel storage and transportation. Receiving diesel deliveries in the tanks, storing fuel and shipping the product to final costumers are activities used to develop and solve optimization models for scheduling.

The mathematical model developed for this system aims to find a sequence that meets the limitations and the request for minimal cost of diesel handling and storage. This problem was modelled using the arrangement of the tanking park, activities restrictions and diesel oil demand as the input parameter data and activities management as the output parameter data. Figure 1 shows the relationship between the parameters mentioned above and the mathematical model.

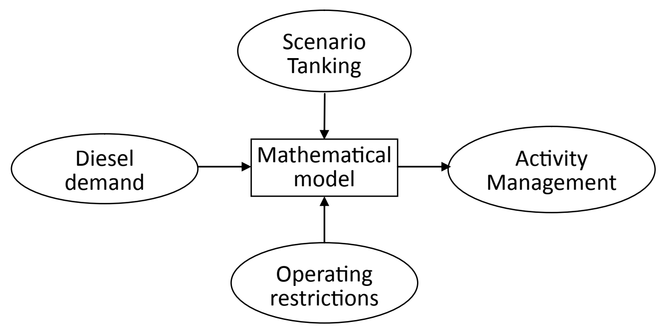

Figure 1. Mathematic Model – Input and Output
Source: Barboza (2005)

The following assumptions and operational constraints were considered in the problem of modeling:

- Diesel volumes stored in the tanks are known;
- The tanks are dedicated, i.e. store a single type of diesel;
- All tanks must be available during the planning horizon, i.e. they must not have maintenance or any operation that would prohibit its use;
- Transition times between tasks are not considered because they are negligible in relation to other operations;
- Each of the tanks can not perform charging and discharging operations simultaneously;
- Each tank can receive from a single source at a given time interval;
- Each tank can send to a single destination at a given time interval;
- The operational constraints of minimum/maximum flow of receipt and dispatch of products must be respected;
- The tanks may not be with volume below the minimum nor above the stipulated maximum;

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

- When a tank starts the load receipt, you must do so to it total filler;
- After filling in, the tank should stand still for a minimum period stipulated for rest and analysis of the stored product. After this time the tank will be available for sending;
- The whole lot required by the customer must be pumped continuously;
- The receiving and sending demands can be met at any time of the planning horizon.
- For the resolution of the model the following techniques were used: Branch and Bound, Simplex Method and GA.

4. MODEL MIXED INTEGER LINEAR PROGRAMMING

The first mathematical model for the problem of transfer and storage using the MILP modeling with uniform discretization in time. continuous, integer and binary variables are used. The binary variables represent decisions to be taken. Any event must start and end at the beginning of each time interval. As mentioned above, the purpose of this model is to minimize operating costs. The restrictions were built in compliance with the operating procedures, the physical constraints of the process and demand.

Nomenclature used in the Model

Indexes:

tq	-	represents the tank number: $tq = 1,2,...,TQ$
c	-	represents the customer number: $c = 1,2,...,C$
t	-	represents the time interval number: $t = 1,2,...,T$

Sets:

CTQ	-	set formed by tanks *tq*
CC	-	set formed by customers *c*
CT	-	set formed by time interval *t*

It is worth noting the use of subscripts to represent the tanks, the customers or the time intervals involved in the parameters and variables.

Parameters:

CB_c	-	pumping cost for the C customer (monetary units / thousand m3);
CA_{tq}	-	storage cost in *tq* tank (monetary units / thousand m3);
CTR_{tq}	-	exchange of *tq* tank cost that receives production (monetary units);
$QRmin$	-	minimum flow received by tanks *tq* (thousands m^3/h);
$QRmax$	-	maximum flow received by tanks *tq* (thousands m^3/h);
$QEmin_c$	-	minimum flow of dispatch to customer c (thousand m3 / h);
$QEmax_c$	-	maximum flow of dispatch to customer c (thousand m3 / h);
$Volmin_{tq}$	-	minimum capacity allowed in the tank *tq* (thousand m^3);
$Volmax_{tq}$	-	maximum capacity allowed in the tank *tq* (thousand m^3);
$Volini_{tq}$	-	volume in the tank *tq* at the beginning of process (thousand m^3);
DEM_c	-	customer *c* diesel demand (thousand m^3).

Binary Variables:

$$Rec_{tq,t} = \begin{cases} 1, \text{if the tank } tq \text{ receives diesel from production in time interval } t \\ 0, \text{otherwise} \end{cases}$$

$$Env_{tq,c,t} = \begin{cases} 1, \text{if the tank } tq \text{ sends to the customer } c \text{ over the time interval } t \\ 0, \text{otherwise} \end{cases}$$

$$XI_{c,t} = \begin{cases} 1, \text{if the customer } c \text{ begin to receive diesel over the time interval } t \\ 0, \text{otherwise} \end{cases}$$

$$XF_{c,t} = \begin{cases} 1, \text{if the customer } c \text{ ends to receive diesel over the time interval } t \\ 0, \text{otherwise} \end{cases}$$

$$TR_{tq,tq',t} = \begin{cases} 1, \text{if the thank } tq \text{ ends to receive and the thank } tq' \text{ begin to receive over the time interval } t \\ 0, \text{otherwise} \end{cases}$$

Continuous Variables:

$QR_{tq,t}$	diesel volume received by the tank *tq* over the time interval *t*;
$QE_{tq,c,t}$	diesel volume sent by tank *tq* to customer *c* over time interval *t*;
$Vol_{tq,t}$	diesel volume stocked in tank *tq* over time *t*;
TI_c	it marks the strt time of receipt of the client *c*;
TF_c	It marks the end time of receipt of the client *c*;

Objective Function:

For this model it is assumed that the optimum operation of the transfer and storage problem is that which minimizes

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

the process operating costs, defined by pumping shipping costs added to the product cost of storage in the tanks and finally added to the cost of transition by exchanging tanks at the receiving process operation.

$$Costs = \sum_{tq \in CTQ} \sum_{c \in CC} \sum_{t \in CT} CB_c.QE_{tq.c.t} + \sum_{tq \in CTQ} \sum_{t \in CT} CA_{tq}.Vol_{tq,t} + \sum_{tq \in CTQ} \sum_{tq' \neq tq \in CTQ} \sum_{t \in CT} CTR_{tq}.TR_{tq,tq',t} \quad (5)$$

Restrictions:

One tank can not receive and send diesel at the same time. Therefore, there are three possible states for the tank: receiving, sending or idle.

$$Rec_{tq,t} + \sum_{c \in CC} Env_{tq,c,t} \leq 1 \qquad \forall\, tq = 1,\ldots,TQ;\; t = 1,\ldots,T \quad (6)$$

As the product flow of the process is continuous, there will always be a tank for receiving diesel each time interval.

$$\sum_{tq \in CTQ} Rec_{tq,t} = 1 \qquad \forall\, t = 1,\ldots,T \quad (7)$$

Only one tank will be able to send diesel to a certain customer over each time interval.

$$\sum_{tq \in CTQ} Env_{tq,c,t} \leq 1 \qquad \forall\, c = 1,\ldots,C;\; t = 1,\ldots,T \quad (8)$$

When a customer starts receiving diesel, must do so without interruption. Therefore, will have only one variable of beginning and end equal 1.

$$\sum_{t \in CT} XI_{c,t} \leq 1 \qquad \forall\, c = 1,\ldots,C$$

and

$$\sum_{t \in CT} \left(XI_{c,t} - XF_{c,t} \right) = 0 \qquad \forall\, c = 1,\ldots,C$$

(9) and (10)

The start time and end of pumping are stored in the auxiliary variables TI_c e TF_c.

$$\sum_{t \in CT} XI_{c,t} \leq 1 \qquad \forall\, c = 1,\ldots,C$$

and

$$\sum_{t \in CT} \left(XI_{c,t} - XF_{c,t} \right) = 0 \qquad \forall\, c = 1,\ldots,C$$

(11) and (12)

The receiving flow should be between maximum and minimum flow stipulated if variable $Rec_{tq,t}$ is equal to "1". If value of variable $Rec_{tq,t}$ is "0", the flow $QR_{tq,t}$ is also equal to "0".

$$QR\,min.Rec_{tq,t} \leq QR_{tq,t} \leq QR\,max.Rec_{tq,t} \qquad \forall\, tq = 1,\ldots,TQ;\; t = 1,\ldots,T \quad (13)$$

The sending flow should be between minimum and maximum flow stipulated if variable $Env_{tq,c,t}$ is equal to "1". If value of variable $Env_{tq,c,t}$ is "0", the flow $QE_{tq,c,t}$ is also equal to "0".

$$QE\,min_c.Env_{tq,c,t} \leq QE_{tq,c,t} \leq QE\,max_c.Env_{tq,c,t} \qquad \forall\, tq = 1,\ldots,TQ;\; c = 1,\ldots,C;\; t = 1,\ldots,T \quad (14)$$

The volume of a tank at a certain time interval shall be equal to incial volume, plus the volume received from the process less the volume sent to customers until this time interval.

$$Vol_{tq,t} = Volini_{tq} + \sum_{\substack{t' \in CT \\ t' \leq t}} \left(QR_{tq,t'} - \sum_{\substack{c \in CC \\ t' \leq t}} QE_{tq,c,t'} \right) \qquad \forall\, tq = 1,\ldots,TQ;\; t = 1,\ldots,T \quad (15)$$

The tanks volume shall be always between minimum and maximum volume determined.

$$Vol\,min_{tq} \leq Vol_{tq,t} \leq Vol\,max_{tq} \qquad \forall\, tq = 1,\ldots,TQ;\; t = 1,\ldots,T \quad (16)$$

The customers demand should be fullfiled in it's entirety.

$$\sum_{tq \in CTQ} \sum_{t \in CT} QE_{tq,c,t} = DEM_c \qquad \forall\, c = 1,\ldots,C \quad (17)$$

The transition occurs when in a time interval a certain tank is receiving from the process and in the following interval another tank is receiving.

$$TR_{tq,tq',t} \leq Rec_{tq,t-1} \qquad \forall\, t = 2,\ldots,T;\; tq, tq' = 1,\ldots,TQ,\; tq \neq tq' \quad (18)$$

$$TR_{tq,tq',t} \leq Rec_{tq',t} \qquad \forall\, t = 2,\ldots,T;\; tq, tq' = 1,\ldots,TQ,\; tq \neq tq' \quad (19)$$

$$TR_{tq,tq',t} \geq Rec_{tq,t-1} + Rec_{tq',t} - 1 \qquad \forall\, t = 2,\ldots,T;\; tq, tq' = 1,\ldots,TQ,\; tq \neq tq' \quad (20)$$

The following restrictions makes the variable $XI_{c,t}$ assumes value "1" if in the interval (t-1) the customer is not receiving and begins to receive in the interval t and assumes value "0" for any other situation.

$$XI_{c,t} \leq \sum_{tq \in CTQ} Env_{tq,c,t} \qquad \forall\, t = 2,\ldots,T;\; c = 1,\ldots,C \quad (21)$$

$$XI_{c,t} \leq 1 - \sum_{tq \in CTQ} Env_{tq,c,t-1} \qquad \forall\, t = 2,\ldots,T;\; c = 1,\ldots,C \quad (22)$$

$$XI_{c,t} \geq \sum_{tq \in CTQ} Env_{tq,c,t} - \sum_{tq \in CTQ} Env_{tq,c,t-1} \qquad \forall\, t = 2,\ldots,T;\; c = 1,\ldots,C \quad (23)$$

The following restrictions makes the variable $XF_{c,t}$ assumes value "1" if in the interval (t - 1) the customer is receiving and begins to not receiving in the interval t and assumes value "0" for any other situation.

$$XF_{c,t} \leq \sum_{tq \in CTQ} Env_{tq,c,t-1} \qquad \forall\, t = 2,\ldots,T;\, c = 1,\ldots,C \quad (24)$$

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

$$XF_{c,t} \le 1 - \sum_{tq \in CTQ} Env_{tq,c,t} \qquad \forall\, t = 2,\ldots,T;\;\; c = 1,\ldots,C \tag{25}$$

$$XF_{c,t} \ge \sum_{tq \in CTQ} Env_{tq,c,t-1} - \sum_{tq \in CTQ} Env_{tq,c,t} \qquad \forall\, t = 2,\ldots,T;\; c = 1,\ldots,C \tag{26}$$

The restrictions (21) to (26) do not contemplate the first time interval. If at beginning of period a tank is pumping, this will be the start of pumping. No pump may close in the first interval.

$$XI_{c,1} = \sum_{tq \in CTQ} Env_{tq,c,1} \qquad \forall\, c = 1,\ldots,C \tag{27}$$

and

$$XF_{c,1} = 0 \qquad \forall\, c = 1,\ldots,C \tag{28}$$

5. MODELING AND METHODOLOGY OF GASSH - MLIP

The model used for the problem is generated from the MILP model with uniform discretization of time (Section 4). For the resolution, this model MILP is modified to be solved by PL. This change includes the removal of some restrictions and binary variables, and linear relaxation of the remaining binary and integer variables. Some variables removed from the model are handled by a Genetic Algorithm Steady State Hybrid (GASSH) and other through a procedure. The values are entered as data in the model LP, which is then resolved by 8.0 LINGO application. The value of the resulting objective function is used as the value of the aptitude function for GASSH. After finishing the process of GASSH, the result is entered as data in the model in MILP. This model is then solved by LINGO 8.0 using the branch and bound technique. Figure 2 shows a simplified flowchart of this modeling.

The variables $Rec_{tq,t}$, $tq = 1,\ldots,TQ$, $t = 1,\ldots,T$ They were treated by GASSH. Since this variable is binary, the individuals of the population of GASSH were formed by randomly generating vector composed of binary digits "0" and "1". A procedure was developed to generate these chromosomes in order to satisfy the restriction (7) which requires that there will always be one and only one tank receiving the output in each time interval. Therefore, for an individual construction should have in each time interval $t \in \{1,\ldots,T\}$, one and only one tank $tq \in \{1,\ldots,TQ\}$ with binary variable value equal "1". As the binary variables $TR_{tq,tq',t}$ with $tq, tq' \in \{1,\ldots,TQ\}$ and $t \in \{1,\ldots,T\}$ depends exclusively from variables $Rec_{tq,t}$, a procedure was implemented to find them, satisfying the restrictions (18), (19) and (20). The procedure verifies, for all tank combinations two by two, if the $Rec_{tq,t}$ variables in the time interval t and (t – 1) are equal "1". If so, the $TR_{tq,tq',t}$ variable will have value "1" in the interval t. For any other situation, the variable takes the value "0". This check is made for the time intervals $t \in \{2,\ldots,T\}$.

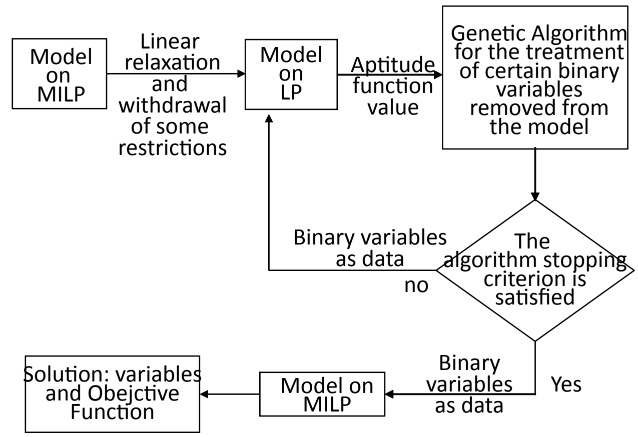

Figure 2. Fluxogram of MILP – Evolucionary Computation

Source: Barboza (2005)

To find the value of the aptitude function, the following steps are followed: chromosome data extraction ($Rec_{tq,t}$, *tq = 1,..., TQ, t = 1,...,T variables*); application of the procedure to generate TRtq variables tq ', t; insertion of data in the LINGO PL 8.0 model; Resolution of this model using the simplex method and obtaining the objective function value to be used as the aptitude function value.

The AGSSH implemented showed the necessity of including a more aggressive local search after a number of iterations in order to get out of minimum local. This search occurs after a certain number of iterations, considered as variable parameter of GASSH. This search consists of selecting an individual of the population and comprehensive exchange of variable values $Rec_{tq,t}$. The individual generated by the change to get better results for the aptitude function will be inserted in the population.

The operators of selection, crossover and mutation used in GASSH are:

Operator of selection: suggested by Mayerle (1994) using the following formula:

$$Select(R) = \left\{ r_j \in R \,/\, j = m + 1 - \left\lceil \frac{-1 + \sqrt{1 + 4.rnd.(m^2 + m)}}{2} \right\rceil \right\} \tag{29}$$

where: R is the set of chromossomes m; r_j is the chromossome j-eth; rnd $\in [0,1)$ is a random number uniformly distributed; $\lceil x \rceil$ is the function that s the function that returns the smallest integer greater than x.

Crossover: of two points. This crossing was specialized for the configuration of the chromosomes involved, in order to generate new viable individuals compared the restriction (7).

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

Mutation: simple mutation. A time interval is randomly chosen for your binary string is modified. Draws then a tank, with the restriction that its value should be "0". Change this value to "1" and the tank that was "1" becomes "0".

Srinivas and Patnaik (1994) adaptive rates were applied for the probability of recombination (crossing) p_c and the probability of mutation p_m, in order to avoid premature convergence of GA for a great location. To minimize problems, consider f' as the lowest value of aptitude among individuals chosen for parents, f_{min}, the lowest value of aptitude and f_{max} the highest value of aptitude, respectively, among all individuals in the population. The formulas (30) and (31) calculate the crossover and mutation probabilities, respectively.

$$p_c = [(k_2 - k_1).f' + f_{max}.k_1 - f_{min}.k_2]/(f_{max} - f_{min}) \quad (30)$$

$$p_m = [(k_4 - k_3).f' + f_{max}.k_3 - f_{min}.k_4]/(f_{max} - f_{min}) \quad (31)$$

The parameters k_1, k_2, k_3 e k_4 are real numbers in the interval $[0,1]$ with $k_1 < k_2$ e $k_3 < k_4$.

The pseudocode of genetic algorithm used in this method is shown below:

Algorithm GASSH

Initialize the population *P* with chromosomes *N*
Evaluate individuals in population *P*
Order the P population in ascending order by the aptitude value
Repeat
  If the operator application of recombination with probability p_c is true **so**
    Select two individuals in *P*
      Apply the recombination operator
      Evaluate the generated individuals
      Insert those individuals in *P* according with their aptitude
    If the operator application of mutation with probability p_m is true **so**
Select one individual in *P*
      Apply the mutation operator
      Evaluate the generated individual
      Insert this individual in *P* according with your aptitude
    If the iterations number for hybridization was achieved, **then**
Make local search
Up to a maximum of generations
End

6. IMPLEMENTATION AND RESULTS

The results are displayed for models with discrete time representation with the implementation of methodologies: Branch and Bound algorithm using the LINGO 8.0 application and GASSH - MILP. The LINGO 8.0 computer application used was provided by the Graduate Program in Numerical Methods in Engineering, Federal University of Parana. The implemented computer programs were developed with Visual Basic 6.0 application (VB 6.0) in the Enterprise Edition. The implementation of GASSH approach - MILP we used two modules in VB, provided by LINDO Systems Inc., for integration between VB 6.0 and 8.0 LINGO. For the described model were made two groups of different tests to obtain results: model resolution in MILP using LINGO 8.0 application and implementation of GASSH - MILP methodology.

The used data for these groups was:

- Tanks number: *TQ* = 4;
- Customers number: *C* = 2;
- Time intervals number: *T* = 24;
  - Pumping cost for the customers: CB_1 = 0,15 monetary units/thousand m^3 and CB_2 = 0,2 monetary units/thousand m^3 of product shipped;
  - Tank storage cost: CA_{tq} = 0.01 monetary units / thousand m3 of product stored in any tank;
  - Exchange cost: CTR_{tq} = 2,00 monetary units for each exchange made between tanks on the receipt operation;
- Minimum flow of receipt by the tanks: *QRmin* = 0,6 thousand m^3/h;
- Maximum flow of receipt by the tanks: *QRmax* = 0,7 thousand m^3/h;
- Minimum flow sent to the customer *c*: $QEmin_1$ = 0,5 thousand m^3/h and $QEmin_2$ = 0,9 thousand m^3/h;
- Maximum flow sent to the customer *c*: $QEmax_1$ = 0,6 thousand m^3/h and $QEmax_2$ = 1 thousand m^3/h;
- Minimum volume allowed in the tanks: $Volmin_{tq}$ = 1 thousand m^3 for all the tanks;
- Maximum volume allowed in the tanks: $Volmax_{tq}$ = 16 thousand m^3 for all the tanks;
  - Volume in the tank *tq* early in the process: $Volini_1$ = 7 thousand m^3, $Volini_2$ = 1 thousand m^3, $Volini_3$ = 1 thousand m^3 and $Volini_4$ = 1 thousand m^3;
- Diesel demand for each customer *c*: DEM_1 = 5 thousand m^3 *and* DEM_2 = 6 thousand m^3;

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

6.1 MILP modelling results with discrete time representation

The above data applied to the MILP model resulted in 390 continuous variables, 766 integers and a total of 2149 restrictions. The model was run 25 times by LINGO 8.0, to meet the great value of 6,285 monetary units. Table 1 shows the values for the average and standard deviation for the number of iterations performed by LINGO 8.0 and the time spent in the process.

Table 1. MLIP model results

Statistics	Nº of Iterations	Computational Time (s)
Média	743124,8	1359,4
Desvio Padrão	273218,7	480,5

Source: Barboza (2005)

6.2 Results for the GASSH – MILP methodology

To obtain the results for the GASSH-MILP model were performed 135 rounds of testing, varying the parameters: population size and the required number of iterations to perform GASSH local search (hybridization). Data on the number of tanks, customers and time intervals shall be in accordance with the model in LP LINGO 8.0, connected to the program. In each test they were saved all individuals with value of aptitude function lower then 6.9, and this value was 10% higher than the optimum value (resolution LP) of 6.266523 monetary units.

In rounds testing was combined the following parameters:

- Population Size: 40, 45 e 50;
- Number of iterations of local search (hybridization): 100, 115 e 125.
  - For adaptive probabilities was used values of k_1 = 0,5, k_2 = 1, k_3 = 0,1 and k_4 = 0,5.

Table 2 show descriptive statistics of GASSH iterations number, LINGO 8.0 iterations number, computational time and aptitude function obtained from the results, separated by intervals to the value of aptitude function *f*.

The result for the overall average of computational time in minutes was 16 minutes and 59 seconds with a standard deviation of 8 minutes and 19 seconds. Through the average of LINGO iterations number and the computational time of the developed program calculated the performance of 1,006 LINGO iterations per second.

Table 2. Descriptives statistics of GASSH – MILP methodology testing (Average ± standard deviation)

Function gaps f	Nº of GASSH Iterations	Nº of LINGO Iterations	Computational Time (s)	Aptitude Function Value
f = 6,285	908 ± 502	1229718 ± 623219	1219 ± 621	6,285 ± 0
6,285 < f ≤ 6,47	679 ± 261	928961 ± 347005	921 ± 352	6,353 ± 0,056
6,47 < f ≤ 6,66	634 ± 241	860666 ± 307462	856 ± 313	6,530 ± 0,025
6,66 < f ≤ 6,9	601 ± 271	837905 ± 365698	834 ±372	6,790 ± 0,071

Source: BARBOZA (2005)

Table 3 shows the average number of iterations GA, LINGO iterations number and computational time for each combination of parameters population size and number of iterations for hybridization when the optimum result of 6,285 was obtained. The improved performance can be observed in population size equals 45 and number of iterations for hybridization equal to 125.

Table 3. Average for the great result of the GA-MILP methodology

Population Size	Nº of Hybridization Iterations	Nº of GA Iterations	Nº of LINGO Iterations	Computational Time (seconds)
40	100	1211	1626152	1609
40	115	971	1306102	1291,7
40	125	940	1219458	1207,2
45	100	786	1100281	1091,5
45	115	845	1152047	1185,3
45	125	732	993706	978,6
50	100	735	1056174	1032,9
50	115	1079	1440459	1428,9
50	125	873	1173083	1149,5
Median	-	908	1229718	1219,4

Source: Barboza (2005)

Figure 3 show the behavior of GASSH iterations number compared to the aptitude function value.

To illustrate the number of LINGO 8.0 iterations, compared to the aptitude function value, the graph on Figure 4 was constructed.

Finally, Figure 5 illustrates the computational time (in seconds) compared to aptitude value.

Observing Figures 3, 4 and 5, we can see a similarity in the behavior of the results for number of GASSH iterations, number of LINGO iterations and computational

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

time with respect to the aptitude function value. Moreover, these graphs show a marked variability and are observed local maximum points, close to 6.76; 6.55 and 6.28. At these points, when the algorithm has reached local minima, a greater number of GASSH iterations, more iterations LINGO and greater computational time has been spent, generating a greater computational effort to continue the search.

An analysis was performed in order to investigate the influence of population size and iterations number for hybridization on the results of GA iterations number, LINGO iterations number and computational time when it reach great result.

Initially, it estimated the Pearson correlation coefficients between the variables number of iterations of the GA, number of LINGO iterations and computational time, two at a time and tested the null hypothesis of no correlation versus the alternative hypothesis of correlation. The results were considered statistically significant at the 0.05 level. Correlation coefficients *r* showed a strong association between the number of iterations of the GA and the number of iterations LINGO ($r = 0.9913$) between the number of iterations of the GA and computational time ($r = 0.9875$) and between number of iterations of LINGO and computational time ($r = 0.9978$). Therefore, analysis was made only for the number of iterations GA.

Considering the levels 40, 45 and 50 for the population size factor and the levels 100, 115 and 125 for the number of iterations for hybridization factor, we performed an analysis of variance to evaluate the influence of the factors on the average number GA iterations to achieve the optimum result. Initially, we tested the null hypothesis of no interaction between the population size and number of iterations for hybridization. The result indicated that there is not this interaction ($p = 0.2307$). Then, when testing the null hypothesis of equal average for the three levels of iterations for hybridization, it was found that there is no significant difference ($p = 0.5992$). Similarly, for the three levels of the population, there was no significant difference in the number of GA iterations ($p = 0.0553$). However, for a significance level of 0.05, it can be said that for the size of the population there is a tendency to statistically significant difference between levels 40, 45 and 50 of the population, compared to the average number of GA iterations to achieve the optimal result.

The similar behavior of the number of GA iterations, number of LINGO iterations and computational time for the population size combinations and number of iterations for hybridization can be observed in the graphs of Figures 6, 7 and 8, constructed from Tables 4, 5 and 6, of the average obtained with all the results for each of these variables. The graphs confirm the results of the correlation coefficient. The improved performance can be observed in population size equals 45 and number of iterations for hybridization equal to 100 (Tables 4, 5 and 6 and Figures 6, 7 and 8).

Table4. Average for number of GASSH iterations according with the number of iterations for hybridization and population size

	Iterations number for hybridization		
Population size	**100**	**115**	**125**
40	834	805	710
45	640	746	696
50	661	979	901

Source: Barboza (2005)

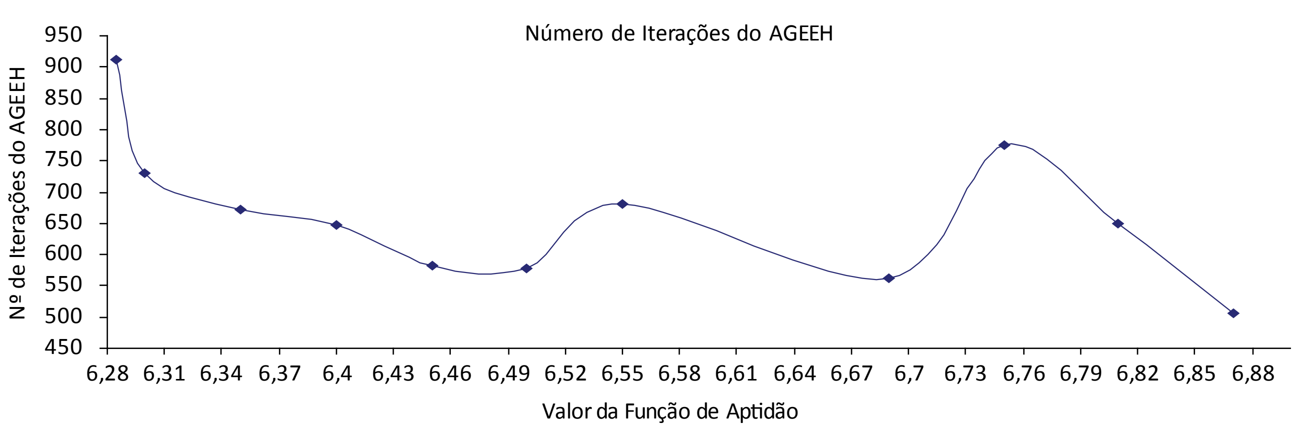

Figure 3. GASSH iterations number compared to aptitude function value

Source: Barboza (2005)

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

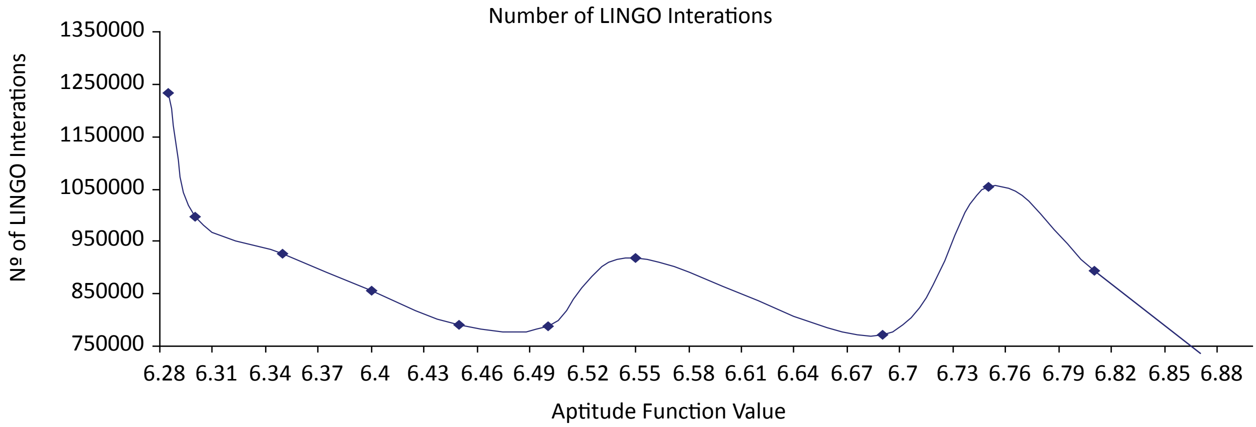

Figure 4. Number of LINGO Iterations compared to aptitude function value

Source: Barboza (2005)

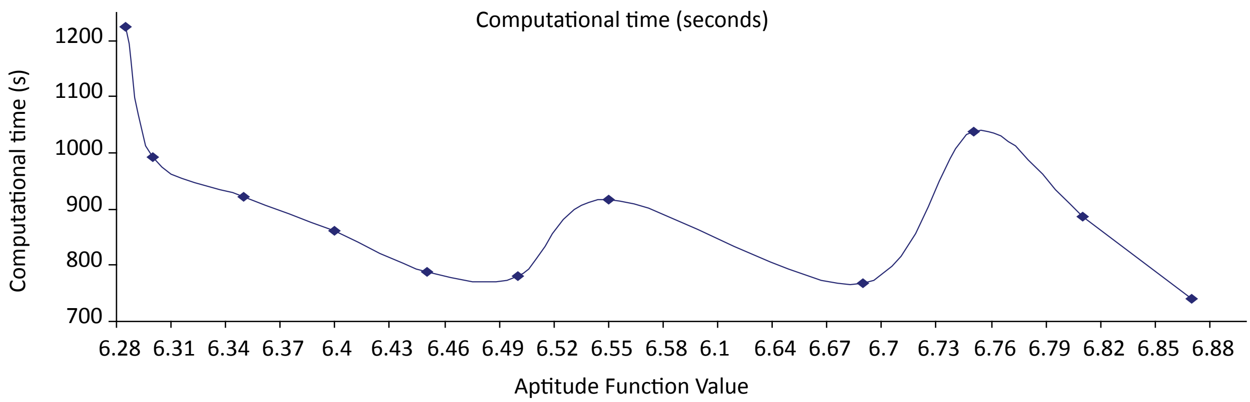

Figure 5. Computational time according aptitude function value

Source: Barboza (2005)

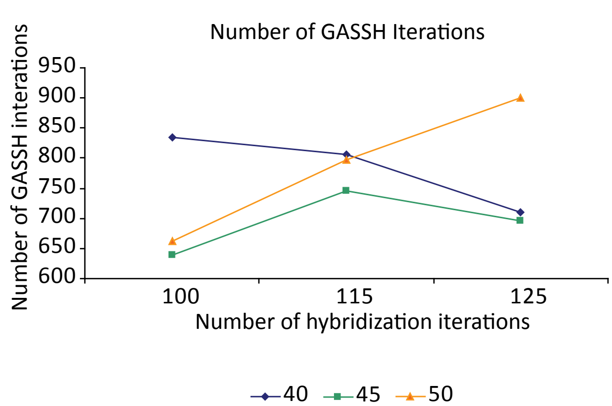

Figure 6. Number of GASSH iterations according with number of iterations for hybridization and population size

Source: Barboza (2005)

Table 5. Average for number of LINGO iterations according with number of iterations for hybridization and population size

	Iterations number for hybridization		
Population Size	**100**	**115**	**125**
40	1142776	1094358	941442
45	907578	1013426	944055
50	941944	1085100	1200496

Source: Barboza (2005)

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

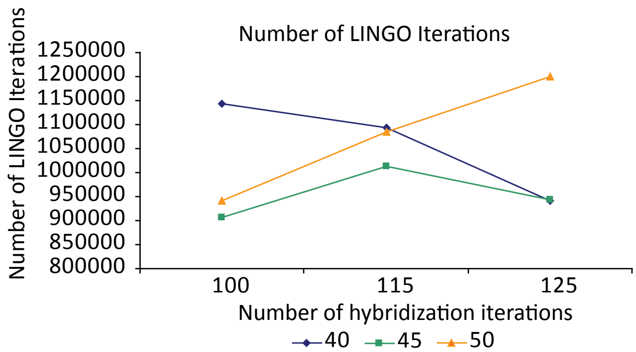

Figure 7. Number of LINGO iterations according with number of iterations for hybridization and population size

Source: Barboza (2005)

Table 6. Average for computational time according with number of iterations for hybridization and population size

	Iterations number for hybridization		
Population size	**100**	**115**	**125**
40	1134	1079	932
45	901	1039	939
50	918	1069	1184

Source: Barboza (2005)

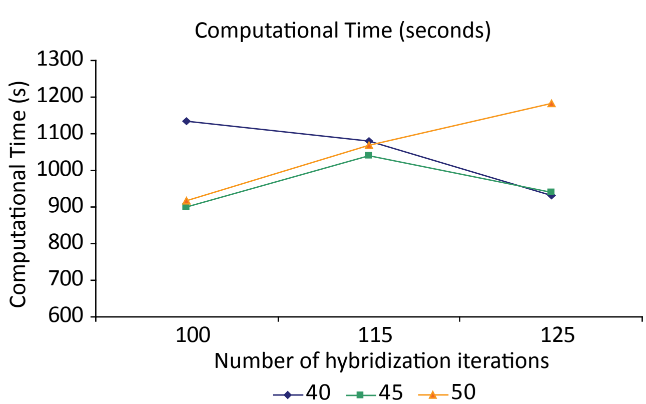

Figure 8. Computational time according with number of iteration for hybridization and population size

Source: Barboza (2005)

7. ANALYSIS OF RESULTS AND CONCLUSIONS

In this work were discussed applications for production scheduling optimization (scheduling) in an oil refinery. The problem addressed was the Diesel transfer and storage, which includes sending diesel tanks and the subsequent distribution of this product to customers. The development and implementation of new methodologies aimed at obtaining efficient solutions that prioritize the financial return in order to become economically viable its practical results.

The process of diesel oil transfer and storage of oil refinery was analysed in order to obtain a model with discrete representation in time with linear characteristics. Being a model MILP with the time discretization, a problem arises in the resolution by LINGO 8.0 application is that reducing the interval size implies a number of intervals (T) higher and thus a larger number of binary variables, so that the computational time increases. This increase in time is due to the fact of the matter is the combinatorial type. In this case, increasing the number of integer variables increases the number of possible combinations for the solution vector, which can become unaffordable problem in terms of computational time, in the resolution with branch and bound algorithm.

For MILP model with the application of LINGO 8.0 were performed 25 tests that resulted in the great value of 6,285 monetary units to the objective function. Table 4 shows the results obtained for the number of iterations and computational time (seconds). The average for the number of iterations was 743,125 with a standard deviation of 273,218.69. For the time, the average was 22 minutes and 39 seconds with a standard deviation of 8 minutes and 52 seconds.

Table 7 shows the means of LINGO 8.0 iterations and computational time of GASSH-MILP and MILP methodologies according to the intervals of the aptitude function (f). Based on these data comparisons were made between the two methodologies regarding the performance.

Table 7. Average summary for MILP and GASSH-MILP methodologies

Function gaps f	Methodology	LINGO Iterations	Computational Time (seconds)
f = 6,285 (resultado ótimo)	PLIM	743125	1359,4
	AGEEH – PLIM	1229718	1219
6,285 < f ≤ 6,47	AGEEH – PLIM	988961	921
6,47 < f ≤ 6,66	AGEEH – PLIM	860666	856
6,66 < f ≤ 6,9	AGEEH – PLIM	837905	834

Source: Barboza (2005)

For optimal results at the GASSH-MILP methodology has an iterations number 65.5% higher and computational time 10.3% lower than in MILP methodology. If $6{,}285 < f \le 6{,}47$, the GASSH-MILP methodology results in a number of iterations 33,1% higher and computational time 32,2% lower. If $6{,}47 < f \le 6{,}66$, GASSH-MILP methodology results in a number of iterations 15,8% higher and computational time 37% lower. And if $6{,}66 < f \le 6{,}9$, GASSH-MILP methodology results in a number of iterations 12.8% higher and computational time 38.6% lower. In summary, for GASSH-MILP methodology, compared with the MILP methodology, the further the optimal value, the lower is the loss compared

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

to the number of iterations and the greater the gain compared to computational time. In general, it is observed that the performance of the GASSH - MILP methodology, regarding computational time was better. It follows, therefore, that the application of the GASSH - MILP methodology, for instance used, should be considered the benefit of a reduced computational time for obtaining a good result, which may not be optimal.

The results obtained for the GASSH-MILP methodology show that the performance depends on the parameters used. From the analysis it can be stated that the number of GA iterations influence more on performance than the number of iterations for hybridization. The parameters used in this study were chosen based on previous testing rounds.

It should be noted that the use of adaptive recombination and mutation probabilities makes the further away an individual is the population's first, higher these probabilities. Thus, this individual ends up being induced to make the recombination and mutation operations. On the other hand, the first individuals in the population will always have lower probability to the chance of these operations is reduced in order to preserve these individuals.

For tests carried out in this work, analyzing the results of Table 3 shows that the best performance in terms of number of iterations of the GA, number of LINGO iterations and computational time to achieve optimum results, it was observed to size population equals 45 and number of iterations for hybridization equal to 125.

The variability of the aptitude function values analyzed in Figures 3, 4 and 5 shows a similar behavior when is observed the number of GASSH iterations, the number of LINGO iterations and computational time. As the value of the function approaches the great value (6.285), this variability decreased and there has been a sharp growth that generates a cost in relation to the performance of GA algorithm. This occurs because the algorithm reaches a local minimum and is in need of an extra effort to get out of this minimum and be directed to the optimal result.

Finally, it is concluded that the GASSH-MILP methodology can contribute significantly to the problem of product transfer and storage in oil refineries. The results showed that the methodology is appropriate, taking into account the reduction of computational time, without much loss in quality solution.

REFERENCES

Almeida, L. F., Pacheco, M. A., Vellasco, M. M. (2003), "Algoritmos Evolucionários na Otimização de Alternativas para o Desenvolvimento de Campos de Petróleo", apresentado no XXIII ENEGEP 2003: Encontro Nacional de Engenharia de Produção, Ouro Preto, MG, 21-24 de Outubro, 2003.

Barboza, A. O. (2005), Simulação e técnicas da computação evolucionária aplicadas a problemas de programação linear inteira mista. Tese de Doutorado em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná, Curitiba, PR.

Carrano, E. G., Takahashi, R. H. C., Cardoso, E. P., Saldanha, R. R., NETO, O.M. (2005), "Optimal Substation Location and Energy Distribution Network Design using a Hybrid GA-BFGS Algorithm", Transmission and Distribution, Vol. 152, No 6, pp. 919-926.

Dantzig, G. B. (1963), Linear Programming and Extensions, Princeton University Press, Princenton, New Jersey, USA.

Duan, Z. et Wang, L. (2011), "Heuristic algorithms for the inverse mixed integer linear programming problem", Journal of Global Optimization, Vol. 51, No. 3, p.463-471.

Floudas, C. A. et Lin, X. (2005), "Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications", Annals of Operations Research, Vol. 139, No. 1, pp. 131-162.

Goldberg, D. E. (1989), Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, INC. USA.

Holland, J. H. (1975), Adaptation in Natural and Artificial Systems, 2 ed., The University of Michigan Press, MI.

ILOG CPLEX 8,0. (2002), User's Manual ILOG SA, Gentilly, France.

Jia, Z. et Ierapetritou, M. (2003), "Mixed-Integer Linear Programming Model for Gasoline Blending and Distribution Scheduling", Industrial & Engineering Chemistry Research, Vol. 42, No. 4, p. 825-835.

Joly, M., Moro, L. F. L., Pinto, J. M. (2002), "Planning and scheduling for Petroleum Refineries using mathematical programming", Brazilian Journal of Chemical Engineering Vol. 19, No. 02, pp. 207-228.

LINDO Systems Inc. (2002), LINDO Systems Client Models.

LINDO Systems, Inc. (2002), "Lingo user's Manual, "LINDO Systems, Chigago, IL.

Liu, G., Zhang, J., Gao, R., Shang, Y. (2009), "An Improved Parallel Adaptive Genetic Algorithm Based on Pareto Front for Multi-Objective Problems", apresentado no 2nd International Symposium on Knowledge Acquisition and Modeling, Wuban, China, 30 de Novembro – 1 de Dezembro, 2009.

Mayerle, S. F. (1994), "Um algoritmo genético para solução do problema do caixeiro viajante", Departamento de Engenharia de Produção e sistemas, Universidade Federal de Santa Catarina, SC.

Electronic Journal of Management & System
Volume 10, Number 4, 2015, pp. 561-574
DOI: 10.20985/1980-5160.2015.v10n4.448

Morabito, R et Abuabara, A. (2008), “Modelos de programação inteira mista para o planejamento do corte unidimensional de tubos metálicos na indústria aeronáutica agrícola”, Gestão & Produção, Vol. 15, No. 3, pp. 605-617.

Moro, L. F. L. (2000), Técnicas de otimização mista-inteira para o planejamento e programação de produção em refinarias de petróleo. Tese de Doutorado, Universidade de São Paulo, SP.

Pinto, J. M. (2000), Planejamento e programação de operações de produção e distribuição em refinarias de petróleo. Tese de Livre Docência, Universidade de São Paulo, SP.

Srinivas, M. et Patnaik, L. M. (1994), “Adaptative Probabilities of Crossover and Mutation in Genetic Algorithms. IEE Transactions on Systems”, Man an Cybernetics, Vol. 24, No. 4, pp. 656-666.

Subbaraj, P., Rengaraj, R., Salivahanan, S. (2011), “Enhancement of Self- Adaptive Real-Coded Genetic Algorithm using Taguchi Method for Economic Dispatch Problem”, Applied Soft Computing, Vol.11, pp. 83-92.

Taha, H. A. (1975), Integer Programming. Theory, Applications, and Computations. Academic Press, Inc. Orlando, Florida, United States of America.

Tanomaru, J. (1995), Motivação, Fundamentos e Aplicações de Algoritmos Genéticos, apresentado no II Congresso Brasileiro de Redes Neurais e III Escola de Redes Neurais, Curitiba, PR, 29 Outubro-01 Novembro, 1995.

Thunberg, J. et Ögren, P. (2011), “A Mixed Integer Linear Programming approach to pursuit evasion problems with optional connectivity constraints”, Autonomous Robots, Vol. 31 No.4, p. 333-343.

Wang, L. et Tang, D. B. (2011), “An improved adaptive genetic algorithm based on hormone modulation mechanism for job--shop scheduling problem”, Expert Systems with Applications, Vol. 38, pp. 7243–7250.

Wu, N., Zhou, M., and Chu, F. (2005), “Short-term Scheduling for Refinery Process: Bridging the Gap between Theory and Applications”, International Journal of Intelligent Control and Systems, Vol. 10, No. 2, pp. 162-174.

Zhang, J.; Shang, Y. (2009), “An Improved Multi-Objective Adaptive Niche Genetic Algorithm Based On Pareto Front”, apresentado no IACC 2009: International Advance Computing Conference, Patiala, 6-7 March, 2009.

Zionts, S. (1974), Linear and Integer Programming. Prentice--Hall, Inc, Englewood Cliffs, New Jersey, USA.