J. Basic. Appl. Sci. Res., 4(2)264-276, 2014

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Genetic Algorithms with Heuristics Rules to Solve Multi Source Single Product Flexible Multistage Logistics Network Problems

Seyed Yaser Bozorgi Rad[1], Mohammad Ishak Desa[2], Majid Firoozi[3]

[1]Islamic Azad University - Babol Branch
[2]Department of Modeling and Industrial Computing, Faculty of Computer Science & Information System. Universiti Teknologi Malaysia (UTM)
[3]Islamic Azad University – Babol branch

Received: November 24 2013
Accepted: January 18 2014

ABSTRACT

To be successful in today's active business competition, enterprises need to design and build effective flexible logistics networks. Since the flexible multistage logistic network (fMLN) problem is NP-hard, many researchers have attempted to use Meta-heuristics methods such as Genetic Algorithms (GAs) to solve the problem. Previous research works using GA for fMLN only considered the problem as a single source, at least in the last network layer between retailer and customer. In real world, however, the problem is one of multi-source logistics network. In this research, the genetic algorithms with penalty method, called P-GA, is used to solve the multi source single product fMLN problem. It is shown however that the P-GA requires unreasonable elapsed time to obtain an acceptable solution. To speed up the algorithm, the research proceeds with the developments of heuristics rules for initialization, crossover and mutation within P-GA and named as HR-GA. This research shows the proposed HR-GA has substantially reduced the elapsed time to obtain better acceptable solution.
KEYWORDS: Flexible Multistage Logistics Network, Genetic Algorithms, Penalty Method, Heuristics Rules GA

1. INTRODUCTION

The structures of the logistics network studied in all the literatures are mostly in the framework of the traditional multistage logistics network (tMLN) model. The logistics network facilities are usually organized into four echelons (*i.e.*, plants, distribution centers (DCs), retailers and customers) in some order, and the product delivery routes should be decided for three stages between every two adjoining echelons sequentially.

The normal delivery in traditional logistics network is from one stage to another adjoining one. Another delivery method is called direct delivery or direct shipment, where goods move from plant to retailer directly and not via distribution centers, or sometimes the customer provides the goods from plant or from distribution center directly and not via retailer. The logistics network with the later delivery method is called flexible Multistage Logistics Network (fMLN) [9]. Figure 1 shows the structure of flexible multi stage logistic network.

The two-stage logistics model is considered to determine the distribution network to satisfy the customer demand at minimum cost subject to the plant and DCs capacity and also the minimum number of DCs to be opened, There are now expanded multi-stage logistics models for applications to real-world cases that include plants, DCs, retailers and customers and this logistics design problem is referred to as three-stages [9]. Paper [9] also claimed that although the traditional Multistage Logistics Network (tMLN) model and its application had made a big success in theory and business practices, the traditional structure of logistics network is unable to fit very well with the fast changing competitive environments and meet the diversified customer demands. Furthermore, paper [9] introduced three new delivery modes in which the goods can move from plants to retailer directly not via distribution centers, or sometimes the customer can get the goods from plant or from distribution center directly not via retailer. The authors called this new logistics network as the flexible Multistage Logistics Network (fMLN).

Paper [15] proposed a hybrid evolutionary algorithm (hEA) and new fuzzy logic control and combine the local search to minimize the summation of transportation cost, inventor holding cost, fixed ordering cost and open cost of facilities. Paper [7] suggested an evolutionary algorithm to determine the optimal combination of shipping alternatives to minimize total logistics costs. Paper [3] realized the artificial neural networks are a very promising tool to predict the short term inter-regional freight distribution. Paper [19] developed a method termed hybrid tabu search, and have applied it to various real-world problems through imposing proper conditions on the generic model. Paper [11] applied mix-integer programming and heuristic solution to integrated logistics model for locating

***Corresponding Author:** Seyed Yaser Bozorgi Rad, Islamic Azad University - Babol Branch. Email: y.bozorgi.r@gmail.com

production and distribution facilities in a multi-echelon environment (Vendors, Plants, Warehouse and Customer). It is single source in last layer (warehouse to customer).

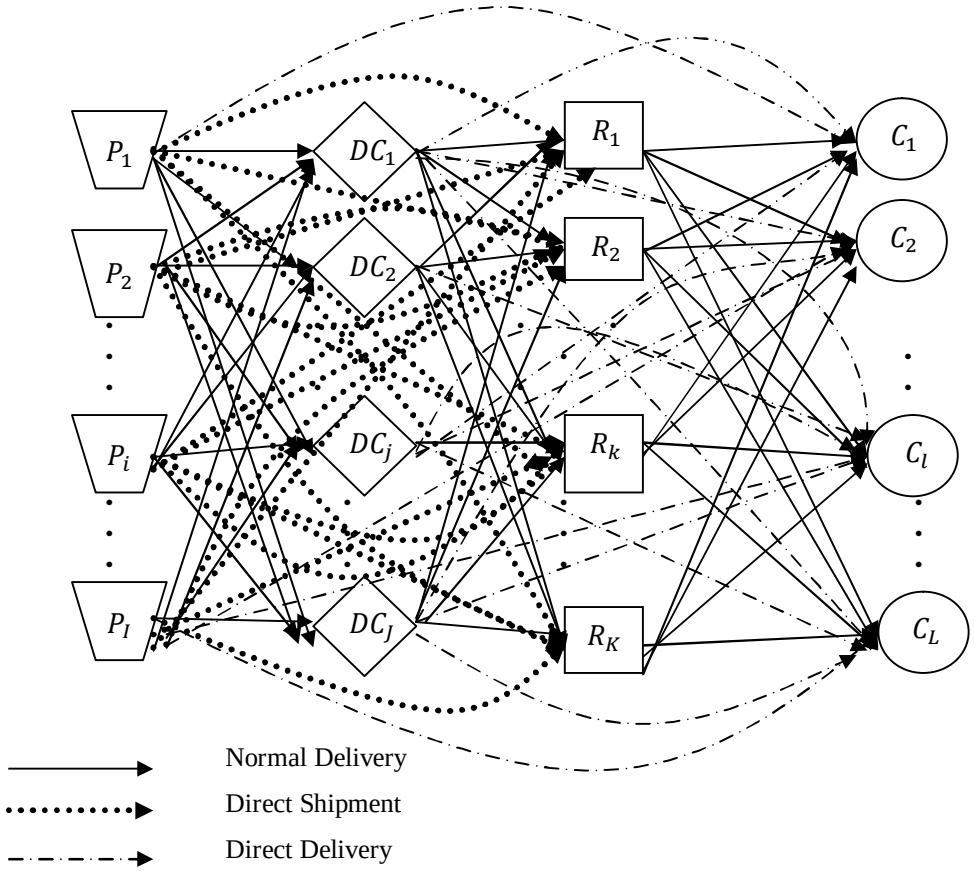

Figure 1: The structure of flexible multistage logistics network (fMLN) models [9].

Paper [18] applied a meta-heuristic method termed hybrid tabu- search to solve the flexible logistics optimization problem under uncertain demand forecasting. Paper [16] applied a mixed integer linear programming to designing a multi echelon supply chain network (SCN) via optimizing commodity transportation and distribution of a SCN.

Recently, GAs has been successfully applied to logistics network models. Paper [4] and Viagnaux and Michalewicz (1991) are among the first who discussed the use of GA for solving linear and nonlinear transportation problems. In their study, while matrix representation was used to construct a chromosome, the matrix-based crossover and mutation had been developed. Another (GA) approach for solving solid TP was given by [13]. They used the three dimensional matrix to represent the candidate solution to the problem. Paper [20] considered production/distribution problem modeled using tsTP and proposed a hybrid genetic algorithm. Paper [8] developed a priority-based Genetic Algorithm (priGA) with new decoding and encoding procedures considering the characteristic of tsTP. The authors of paper [2] extended priGA to solve a single-product multi-stage logistics design problem. The objectives are minimization of the total cost of supply chain, maximization of customer services that can be rendered to customers in terms of acceptable delivery time (coverage), and maximization of capacity utilization balance for DCs (*i.e*. equity on utilization ratios). Furthermore, paper [14] proposed a hybrid genetic algorithm to solve the location-allocation model's problem of logistic network, and the authors of paper [2] also apply the priGA to solve a single-source, multi-product multi-stage logistics design problem. As an extended multi-stage logistics network model, paper [12] apply the priGA to solve a multi-stage reverse logistics network problem (mrLNP), minimizing the total costs to reverse logistics shipping cost and fixed cost of opening the disassembly centers and processing centers. Paper [10] proposed a new approach called spanning tree-based hybrid genetic algorithm (hst- GA) to solve the multi-time period production/distribution and inventory problem (mt-PDI). Paper [5] presented an innovative encoding–decoding procedure embedded within a genetic algorithm (GA) to

minimize the total logistic cost resulting from the transportation of goods and the location and opening of the facilities in a single product three-stage supply chain network.

For any optimization problem, there is an optimization criterion (i.e. evaluation function) to be minimized or maximized. The evaluation function represents a measure of the quality of the developed solution. Searching the space of all possible solution is a challenging task. An additional constraint on the domain of search for the parameters makes the problem quite difficult. The constraints might affect the performance of the evolutionary process since some of the produced solutions (i.e individuals) may be unfeasible. Unfeasible solution represents a waste of computation effort. In fact, it was reported that no general methodology to handle constraints exist although several methods were introduced. Rejecting unfeasible individuals, penalizing unfeasible individuals or moving these individuals to the feasible domain are among the many methods proposed [17].

There are some approaches to handle the constraints optimization problems such as death penalty, static penalties, dynamic penalties, GENOCOP system, Behavioral memory and etceteras [22]. For some similar problems to fMLN with high constraints some researchers tried to add some heuristic rules to GA to satisfy the problem constraints and obtain a good solution. Paper [21] proposed a random search based on heuristic rules and a dynamic rule selection method based on GA to solve large size single-stage batch scheduling problem and paper [1] proposed some heuristic rules embedded GA to solve In-Core Fuel Management Optimization Problem. The author of paper [6] compared three different heuristics based Evolutionary Algorithm (EA) on the same problems and suggested the best one to solve the constraints optimization problems.

The general objective of this paper is to discuss the algorithms that we have developed to solve the multi-source single product flexible Multistage Logistics Network (fMLN) problem.

2. MATERIALS AND METHODS

2.1 Multi Source Flexible Multistage Logistics Network Model

In general, a multi source flexible multistage network (fMLN) problem is to determine the optimum product quantity shipped from plants to the customers and the best product delivery routes to fulfill the customer's order that minimize the total logistics network costs. This research utilizes the fMLN model formulated by [9] as the base model. The model is formulated based on the following assumptions:

1. The single product case of a logistics network optimization problem is considered.
2. A single time period, such as one week or one month is considered.
3. In the logistics network, there are maximum four echelons: plants, DCs, retailers and customers.
4. There are three delivery modes: normal delivery, direct shipment and direct delivery, as mentioned above.
5. Each customer is served by only one facility.
6. Customer demands are known in advance.
7. Customers will get products at the same price, no matter where she/he gets them. It means that the customers have no special preferences [9].

The following notations are used to formulate the mathematical model:

Notation:

Indices:

i:	index of plant $(i = 1,2,3,\cdots,I)$
j:	index of DC $(j = 1,2,3,\cdots,J)$
k:	index of retailer $(k = 1,2,3,\cdots,K)$
l:	index of customer $(l = 1,2,3,\cdots,L)$

Parameters:

I	number of plants
J	number of DCs
K	number of retailers
L	number of customers
P_i	Plant i
DC_j	DC_j
R_k	Retailer k
C_l	Customer l
B_i	Output of plant i

d_l Demand of customer l
C_{1ij} Unit shipping cost of product from P_i to DC_j
C_{2jk} Unit shipping cost of product from DC_j to R_k
C_{3kl} Unit shipping cost of product from R_k to C_l
C_{4il} Unit shipping cost of product from P_i to C_l
C_{5jl} Unit shipping cost of product from DC_j to C_l
C_{6ik} Unit shipping cost of product from P_i to R_k
u_j^D Upper bound of the capacity of DC_j
u_k^R Upper bound of the capacity of R_k
f_j^F Fixed part of the open cost of DC_j
C_j^{1v} Variant part of the open cost (lease cost) of DC_j
q_j^1 Throughout of DC_j
$q_j^1 = \sum_{i=1}^{I} X_{1ij}, \forall j$
f_j Open cost of DC_j
$f_j = f_j^F + C_j^{1v} q_j^1, \forall j$
g_k^F Fixed part of the open cost of R_k
C_k^{2v} Variant part of the open cost (lease cost) of DC_j
q_k^2 Throughout of R_kPiet S
$q_k^2 = \sum_{l=1}^{L} X_{3kl}, \forall k$ Pie Slats Piet A. Slats
g_k= Open cost of R_k
$g_k = g_k^F + C_k^{2v} q_k^2, \forall k$

Decision Variables:

X_{1ij} Transportation quantity from P_i to DC_j
X_{2jk} Transportation quantity from DC_j to R_k
X_{3kl} Transportation quantity from R_k to C_l
X_{4il} Transportation quantity from P_i to C_l
X_{5jl} Transportation quantity from DC_j to C_l
X_{6ik} Transportation quantity from P_i to R_k

$$y_j^1 = \begin{cases} 1, & if\ DC_j\ is\ open \\ 0, & otherwise \end{cases}$$

$$y_k^2 = \begin{cases} 1, & if\ R_k\ is\ open \\ 0, & otherwise \end{cases}$$

The objective function is to minimize the total logistic cost:

$$Min\ Z = \sum_{i=1}^{I}\sum_{j=1}^{J} C_{1ij}X_{1ij} + \sum_{j=1}^{J}\sum_{k=1}^{K} C_{2jk}X_{2jk} + \sum_{k=1}^{K}\sum_{l=1}^{L} C_{3kl}X_{3kl} + \sum_{i=1}^{I}\sum_{l=1}^{L} C_{4il}X_{4il} + \sum_{j=1}^{J}\sum_{l=1}^{L} C_{5jl}X_{5jl} + \sum_{i=1}^{I}\sum_{k=1}^{K} C_{6ik}X_{6ik} + \sum_{j=1}^{J} f_j y_j^1 + \sum_{k=1}^{K} g_k y_k^2 \quad \textbf{(1)}$$

Subject to:

$$\sum_{j=1}^{J} X_{1ij} + \sum_{l=1}^{L} X_{4il} + \sum_{k=1}^{K} X_{6ik} \le b_i, \quad \forall i \quad \textbf{(2)}$$

$$\sum_{i=1}^{I} X_{1ij} = \sum_{k=1}^{K} X_{2jk} + \sum_{l=1}^{L} X_{5jl}, \quad \forall j \quad \textbf{(3)}$$

$$\sum_{j=1}^{J} X_{2jk} + \sum_{i=1}^{I} X_{6ik} = \sum_{l=1}^{L} X_{3kl}, \quad \forall k \quad \textbf{(4)}$$

$$\sum_{i=1}^{I} X_{4il} + \sum_{j=1}^{J} X_{5jl} + \sum_{k=1}^{K} X_{3kl} \ge d_l, \quad \forall l \quad \textbf{(5)}$$

$$\sum_{i=1}^{I} X_{1ij} \le u_j^D y_j^1, \quad \forall j \quad \textbf{(6)}$$

$$\sum_{l=1}^{L} X_{3kl} \le u_k^R y_k^2, \quad \forall k \quad \textbf{(7)}$$

$X_{1ij}, X_{2jk}, X_{3kl}, X_{4il}, X_{5jl}, X_{6ik} \geq 0 \quad , \quad \forall i, j, k, l$ **(8)**

$y_j^1 , y_k^2 \in \{0,1\} , \forall j, k$ **(9)**

Where the objective function in Eq. 1 means to minimize the total logistics cost, here we consider the shipping cost. The constraint in Eq.2 represents the production limit of plants. The constraints in Eq.3 and Eq.4 are due to the flow of conservation principle. The constraint in Eq.5 ensures that the customers' demands will be satisfied. The constraints in Eq.6 and Eq.7 ensure that the upper bound of the capacity of DCs and retailers cannot be surpassed.

Although the authors of paper [9] has included assumption 5, that is, every customer is served by only one facility, which leads to a single source fMLN model, and the above formulated model is in fact a multi source fMLN model. This is obvious from the formulated constraint in Eq. (5). This equation represent that each customer can be served by multi facilities (plants, DCs and retailers) simultaneously. A test using LINDO (a software to solve linear programming problem using simplex method) proved that the above formulation is a multi source fMLN model.

3. RESULT AND DISCUSSION

The algorithm used to solve the above mentioned problem in this research is the proposed heuristics rules for P-GA operators based on GA flexibility. As it was mentioned before, the multi source single product fMLN problem is quite large and complex with a large problem space. Therefore it is hard to obtain an acceptable solution within a reasonable time considering that GA is a random based algorithm. The newly proposed heuristics rules can help P-GA to speed up the running time besides finding better results. In Section 4.5, the proposed heuristics rules and algorithms namely HR-GA and also the numerical result will be presented.

4. Proposed Genetic Algorithm with Heuristics Rules (HR-GA)

The Genetic Algorithm (GAs) using the Darwinian concept was first introduced by Holland (1975). GA is useful when a large search space with little knowledge of the solution is presented, and has been successful in providing near optimal solution for many diverse and difficult problems [10].

4.1 Chromosome Representation

Chromosome representation is one of the important issues that affect the performance of GAs. Usually different problems have different data structures or genetic representations. Tree-based representation is known to be one way of representing network problems [2]. The three ways of encoding tree are: (1) edge-based encoding, (2) vertex-based encoding and (3) edge-and-vertex encoding (Gen and Cheng, 2000). In general, since each customer for all products has to be assigned to only one source (facility) on the last stage of the problem, integer encoding was used to define this situation. The length of the last segment on a chromosome equals the number of customer and the length of chromosome equals to total number of sources (I) and depots (J and K).

Paper [9] has discussed about chromosome representation of GA to solve flexible multistage logistics network. As shown in Figure 4.3, in the genes with length 3×L (L is the total number of customers), every three loci constitute one unit, each of which represents a delivery route to a customer, from the plant via DC and retailer. The allele on first locus of each unit means the ID of a plant, the start node of the delivery path. The second is the ID of DCs, and the last one indicates the ID of retailers on the path.

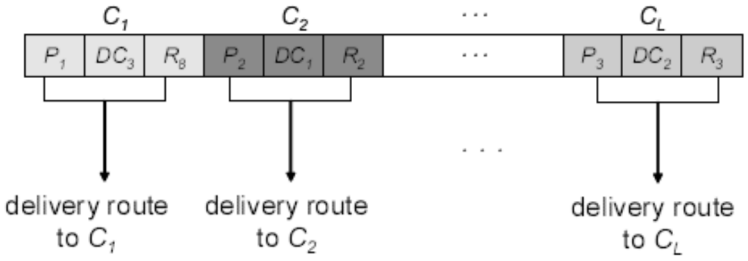

Figure 2: Representation of vertex-based [9].

Based on this model, at least one plant is required in the delivery route to a customer. However, if the total demand to the plant exceeds its supply capacity, the customer is assigned to another plant with sufficient products supply and the lowest transportation price between the plant and the customer.

According to paper [9] chromosome representation for fMLN which is vertex-based the solution is based on finding the best route for delivering the product to each customer when the network is single source in the last layer which is between retailers and customers. In this strategy the customer is not allowed to split the order to fulfill from different sources simultaneously. It is more realistic that every customer can be served by multi facilities. Hence, the chromosome representation of this research is completely different with the previous one. In this part of the research the chromosome representation for multi source single product fMLN problem is considered based on the total number of decision variables which is edge-based. Every gene represents the product amount shipped between different origins. In addition, some of the genes are containing the decision variables for open/close DCs and retailers. Therefore, the solution in this research is based on obtaining the optimum product amount, best product delivery route for each customer and optimum total logistics network cost where customers are allowed to split their orders in order to fulfill more than one resource at the same time by taking into priority the lower cost of each route.

As shown in the fMLN model, there are six types of shipment in flexible multi-stage logistic network. So, the six types of decision variables X_{1ij} , X_{2jk}, X_{3kl} , X_{4il} , X_{5jl} , X_{6ik} where depending on the number of plants, DCs, Retailers and customers, the number of *X* (product amount) will change. Therefore the total number of gene of every chromosome is equal to:

$$I\times\left(J+K+L\right)+J\times\left(K+L\right)+\left(K\times L\right)+J+K$$

Figure 4.4 shows the proposed edge-based chromosome representation in this research to solve the multi source single product fMLN problem.

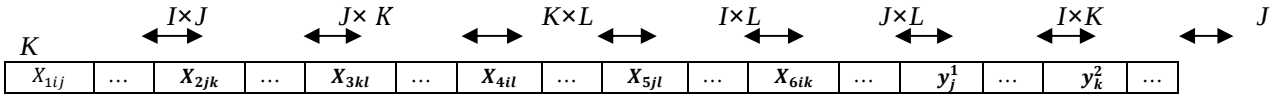

Figure 3: Proposed edge-based chromosome representation

4.2 Initialization

The first step in GA is initialization. In this step the first population is generated randomly based on the defined chromosome representation. The number of chromosome in the population is not fixed and can be determined based on experiment. The first generated population will be improved by the crossover and mutation operators of GA. The value of every gene will be generated randomly within a defined and specific range. In this research it is defined that, randomize (z) is a random integer value between 0 and z inclusive. Here the traditional random initialization method is applied randomly and the product amount of each echelon is assigned with relevant locus as shown in Figure 4.4.

Procedure: *Initialization by P-GA*
Input: *Number of plants (I), DCs (J), Retailers (K), Customers (L), Upper bound of the capacity of DC (UD), Upper bound of the capacity of Retailer (UR) and Customer demand (D)*
Output: *the chromosome* $CH_y[.]$ */* $CH_y[.]$ *is a chromosome/**
Step 0:
for *i=1* ***to*** *I*
for *j=1 to J*
$CH_y[\, i\times J - J + j\,]$ *randomize (UD[j])* */X_{1ij} is defined/**
end *// randomize(z) is a random integer value between 0 and z inclusive //**
end
for *j=1* ***to*** *J*
for *k=1 to K*
$CH_y[\, I\times J + j\times K - K+k]$ *randomize (UR[j])*
*/X_{2jk} is defined/**
end
end
for *k=1* ***to*** *K*
for *l=1 to L*
$CH_y[I\times J + J\times K + k\times L - L+l]$ *randomize (D[j])*
*/X_{3kl} is defined/**
end
end
for *i=1* ***to*** *I*

```
        for l=1 to L
                    CH_y[ I×J + J×K + K×L + i×L – L+l]      randomize (D[j])
                    */X_4il is defined/*
        end
    end
    for j=1 to J
        for l=1 to L
                    CH_y[ I×J + J×K + K×L+ I×L +j×L – L+l]      randomize (D[j])
        end         */X_5jl is defined/*
    end
   for i=1 to I
      for k=1 to  k
                    CH_y[ I×J + J×K + K×L + I×L + J×L + i×K – K+ k]      randomize (UR[j])
      end       */X_6ik is defined/*
       end
       for j=1 to J
                     CH_y[ I×J + J×K + K×L + I×L + J×L + I×K + j ]      randomize (1)

    end           */ y_j^1 is defined/*
       for k=1 to K
                   CH_y[ I×J + J×K + K×L + I×L + J×L + I×K + J + k ]      randomize (1)

    end           */ y_k^2 is defined/*
Step 1: out put the chromosome CH_y[.]
```

Figure 4: Pseudo-code of chromosome generation

According to the multi source single product fMLN model, y_j^1 and y_k^2 are decision variables and related to the chromosome. y_j^1 and y_k^2 are useful in making decisions for every DCs and retailers whether they are to be opened or closed in terms of network cost. In finding the optimal solution, y_j^1 will not be zero when there is shipment between plants and DCs ($X_{1ij} \neq 0$); which means that when there is value of X_{1ij} therefore y_j^1 must not be zero and the specific DC must be opened to store the shipped product. The same applies for shipped product from retailers to customers (X_{3kl}) and the condition of related retailers (open/close).

With the above explanation, the following two constraints are added in the solution of this research to help GA to generate randomly the proper X_{1ij} , X_{3kl} where matched with y_j^1 and y_k^2.

$$y_j^1 = \begin{cases} 1 , & \text{if } \sum_{i=1}^{I} X_{1ij} \neq 0 \ , \ \forall j \\ 0 & , \text{ otherwise} \end{cases} \quad \textbf{(10)}$$

$$y_k^2 = \begin{cases} 1 , & \text{if } \sum_{l=1}^{L} X_{3kl} \neq 0 \ , \forall k \\ 0 & , \text{ otherwise} \end{cases} \quad \textbf{(11)}$$

It is because when it comes to the fitness function in the selection part of GA, the priority of the randomly generated chromosome that shows $X_{1ij} \neq 0$ and $y_j^1 = 1$ is higher than the other similar chromosome which shows $X_{1ij} \neq 0$ and $y_j^1 = 0$, because it is impossible to have optimum solution where $X_{1ij} \neq 0$, $X_{3kl} \neq 0$ and $y_j^1 = 0$, $y_k^2 = 0$.

4.3 Genetic Operators

Crossover and mutation are used to strengthen the search ability of the algorithm. Here the standard two cut points crossover are applied. Two parents are chosen from the population set according to crossover rate and an offspring is created. The parents' chromosomes are then combined into the offspring through a process called "crossover" in which certain genes from each parent are mixed together. An important event for the evolution of any species is "mutation". In this research one gene of the selected offspring chromosomes will be mutated randomly. Mutation is not always required in all mating. Therefore the mutation rate usually is less than the crossover rate and a probability of mutation equation should be set up (this will vary depending on the problem). In order to decide which traits are beneficial and should be passed on; a fitness algorithm must be performed on the children.

In order to decide which traits are beneficial and should be passed on, a fitness algorithm must be performed on the children. These fitness algorithms are completely problem specific.

4.4 Overall Procedure of Genetic Algorithm with Penalty Methods for fMLN

The overall pseudo-code procedure of random product quantity based GA is outlined in the following procedure (Figure 5).

```
procedure: P-GA for fMLN cost optimization
        input: problem parameters and GA parameters
        output: a best solution
  begin
          s ←0; // s: generation number
         initialize P(s) using edge-based encoding; // P(s): The first population
         fitness evaluation(P);
    while (not terminating condition) do
        crossover P(s) to yield O(s) by two-cut point; // O(s): offspring
        mutation P(s) to yield O(s) by insertion mutation;
       Penalty-based fitness evaluation(O);
        select P(t+1) from P(s) and O(s);
           s ← s+1
      end
        output: a best solution
 end
```

Figure 5: The overall pseudo-code for P-GA

5. Proposed Heuristics Rules for P-GA Initialization and Repairing the Crossover Operator

To initiate the population, heuristics rules are used to satisfy the existing constraints because without these rules the probability of satisfying all constraints is low. Besides, it will take a longer time for GA to search and find the relevant value of every gene that satisfies the constraints as the number of gene for a large size problem is too many. Considering the fact that the chance of satisfying all problem constraints by offspring after crossover in GA is low, it is necessary to repair the crossover operator so that the offspring could be kept within the feasible region as well.

In this research, seven specific steps are defined to show the newly proposed embedded rules for GA initiation and repairing the crossover operator that will be used to solve multi source single product fMLN problem and they are as follows:

Step 1: Generate an initial population randomly according to population size.

Step 2: Let $\left[X_{1ij}\right]_{I\times J} = \begin{bmatrix} X_{111} & \cdots & X_{11J} \\ \vdots & \ddots & \vdots \\ X_{1I1} & \cdots & X_{1IJ} \end{bmatrix}$ be the matrix of product amount shipped from P_i to DC_j .

(a) If $\sum_{i=1}^{I} X_{1ij} \neq 0$ and $y_j^1 = 0$ then set $y_j^1 = 1, \forall j$

(b) Calculate $e_j = u_j^D y_j^1 - \sum_{i=1}^{I} X_{1ij}$, $\forall j$.

(c) If $e_j < 0$, add e_j to the genes of X_{1ij} randomly by the following procedures:

(c.1) Calculate $e_j' = \sum_{i=1}^{I} X_{1ij} + e_j$, $\forall j$

(c.2) Generate the new $\left[X_{1ij}\right]_{I\times J}$ denoted as $[X'_{1ij}]_{I\times J} =$

$$\begin{bmatrix} X'_{111} = randomize(e'_1) & \cdots & X'_{11J} = randomize(e'_j) \\ \vdots & & \vdots \\ X'_{1i1} = randomize(e'_1 - \sum_{q=1}^{i-1} X'_{1q1}) & \cdots & X'_{1iJ} = randomize(e'_j - \sum_{q=1}^{i-1} X'_{1qj}) \\ \vdots & & \vdots \\ X'_{1I1} = e'_1 - \sum_{i=1}^{I-1} X'_{1i1} & \cdots & X'_{1IJ} = e'_j - \sum_{i=1}^{I-1} X'_{1ij} \end{bmatrix}$$

where randomize(z) is a random integer value between 0 and z inclusive.

Step 3: Let $[X_{3kl}]_{K\times L} = \begin{bmatrix} X_{311} & \cdots & X_{31L} \\ \vdots & \ddots & \vdots \\ X_{3K1} & \cdots & X_{3KL} \end{bmatrix}$ be the matrix of product amount shipped from R_k to $customer_l$. for k = 1, 2,⋯, K

(a) If $\sum_{l=1}^{L} X_{3kl} \neq 0$ and $y_k^2 = 0$ then set $y_k^2 = 1$.

(b) Calculate $s_k = u_k^R - \sum_{l=1}^{L} X_{3kl}$,$\forall k$

(c) If $s_k < 0$, add s_k to the genes of X_{3kl} randomly by the following procedures: generate the new $[X_{3kl}]_{K\times L}$ denoted as $[X'_{3kl}]_{K\times L}$ using the following procedures:

(c.1) Calculate $s_k' = \sum_{l=1}^{L} X_{3kl} + s_k$, $\forall k$

(c.2) Generate the new $[X_{3kl}]_{K\times L}$ denoted as $[X'_{3kl}]_{K\times L}$=

$$\begin{bmatrix} X'_{311} = randomize(s'_1) & \cdots & X'_{31l} = randomize\,(s'_1 - \sum_{q=1}^{l-1} X'_{31q}) & \cdots & X'_{31L} = s'_1 - \sum_{l=1}^{L-1} X'_{31l} \\ \vdots & & \vdots & & \vdots \\ X'_{3K1} = randomize(s'_K) & \cdots & X'_{3Kl} = randomize\,(s'_K - \sum_{q=1}^{l-1} X'_{3kq}) & \ldots & X'_{3KL} = s'_K - \sum_{l=1}^{L-1} X'_{3kl} \end{bmatrix}$$

Step 4: For i = 1, 2, ⋯, I

(a) Calculate $u_i = b_i - (\sum_{j=1}^{J} X'_{1ij} + \sum_{l=1}^{L} X_{4il} + \sum_{k=1}^{K} X_{6ik})$.

(b) If $u_i < 0$, add u_i arbitrary to the genes of X_{4il} or X_{6ik} randomly by the following procedures: generate the new $[X_{4il}]_{I\times L}$ denoted as $[X'_{4il}]_{I\times L}$ or new $[X_{6ik}]_{I\times K}$ denoted as $[X'_{6ik}]_{I\times K}$ arbitrary using the following procedures:

(b.1) Calculate $u'_i = b_i + u_i, \forall i$

(b.2) Generate the new $[X_{4il}]_{I\times L}$ denoted as $[X'_{4il}]_{I\times L}$ =

$$\begin{bmatrix} X'_{411} = randomize(u'_1) & \cdots & X'_{41l} = randomize\,(u'_1 - \sum_{q=1}^{l-1} X'_{41q}) & \cdots & X'_{41L} = u'_1 - \sum_{l=1}^{L-1} X'_{41l} \\ \vdots & & \vdots & & \vdots \\ X'_{4I1} = randomize(u'_I) & \cdots & X'_{4Il} = randomize\,(u'_I - \sum_{q=1}^{l-1} X'_{4iq}) & \ldots & X'_{4IL} = u'_I - \sum_{l=1}^{L-1} X'_{4il} \end{bmatrix}$$

or

(b.2.1) Generate the new $[X_{6ik}]_{I\times K}$ denoted as $[X'_{6ik}]_{I\times K}$ =

$$\begin{bmatrix} X'_{611} = randomize(u'_1) & \cdots & X'_{61k} = randomize\,(u'_1 - \sum_{q=1}^{k-1} X'_{61q}) & \cdots & X'_{61K} = u'_1 - \sum_{k=1}^{K-1} X'_{61k} \\ \vdots & & \vdots & & \vdots \\ X'_{6I1} = randomize(u'_I) & \cdots & X'_{6Ik} = randomize\,(u'_I - \sum_{q=1}^{k-1} X'_{6iq}) & \ldots & X'_{6IK} = u'_I - \sum_{k=1}^{K-1} X'_{6ik} \end{bmatrix}$$

Step 5: For l= 1, 2, ⋯, L

(a) Calculate $A_l = d_l - (\sum_{i=1}^{I} X'_{4il} + \sum_{j=1}^{J} X_{5jl} + \sum_{k=1}^{K} X'_{3kl})$.

(b) If $A_l > 0$, add A_l to the genes of X_{5jl} randomly by the following procedures:

(b.1) Calculate $A'_l = \sum_{j=1}^{J} X_{5jl} + A_l, \forall l$

(b.2) Generate the new $[X_{5jl}]_{J\times L}$ denoted as $[X'_{5jl}]_{J\times L}$=

$$\begin{bmatrix} X'_{511} = randomize(A'_1) & \cdots & X'_{51L} = randomize(A'_L) \\ \vdots & & \vdots \\ X'_{5j1} = randomize(A'_1 - \sum_{q=1}^{j-1} X'_{5q1}) & \ldots & X'_{5jL} = randomize(A'_L - \sum_{q=1}^{j-1} X'_{5ql}) \\ \vdots & & \vdots \\ X'_{5J1} = A'_1 - \sum_{j=1}^{J-1} X'_{5j1} & \ldots & X'_{5JL} = A'_L - \sum_{j=1}^{J-1} X'_{5jl} \end{bmatrix}$$

Step 6: Let $[X_{2jk}]_{J\times K} = \begin{bmatrix} X_{211} & \cdots & X_{21K} \\ \vdots & \ddots & \vdots \\ X_{2J1} & \cdots & X_{2JK} \end{bmatrix}$ be the matrix of product amount shipped from DC_j to R_k. For j = 1, 2, ⋯, J

(a) Calculate $v_j = \sum_{i=1}^{I} X'_{1ij} - (\sum_{k=1}^{K} X_{2jk} + \sum_{l=1}^{L} X'_{5jl})$.

(b) If $v_j \neq 0$, add v_j to the genes of X_{2jk} randomly by the following procedures:

(b.1) Calculate $v'_j = \sum_{k=1}^{K} X_{2jk} + v_j, \forall j$

(b.2) Generate the new $[X_{2jk}]_{J\times K}$ denoted as $[X'_{2jk}]_{J\times K}$=

$$\begin{bmatrix} X'_{211} = randomize(v'_1) & \cdots & X'_{21k} = randomize(v'_1 - \sum_{q=1}^{k-1} X'_{21q}) & \cdots & X'_{21K} = v'_1 - \sum_{k=1}^{K-1} X'_{21k} \\ \vdots & & \vdots & & \vdots \\ X'_{2J1} = randomize(v'_J) & \cdots & X'_{2Jk} = randomize(v'_J - \sum_{q=1}^{k-1} X'_{2jq}) & \ldots & X'_{2JK} = v'_J - \sum_{k=1}^{K-1} X'_{2jk} \end{bmatrix}$$

Step 7: Let $[X'_{6il}]_{K\times L} = \begin{bmatrix} X'_{611} & \cdots & X'_{61L} \\ \vdots & \ddots & \vdots \\ X'_{6I1} & \cdots & X'_{6IL} \end{bmatrix}$ be the matrix of product amount shipped from R_k to $customer_l$. For k = 1, 2,⋯, K

(a) Calculate $w_k = \sum_{l=1}^{L} X'_{3kl} - (\sum_{j=1}^{J} X_{2jk} + \sum_{i=1}^{I} X'_{6ik})$

(b) If $w_k \neq 0$, add w_k to the genes of X'_{6ik} randomly by the following procedures:

(b.1) Calculate $w'_k = \sum_{i=1}^{I} X'_{6ik} + w_k \; \forall k$

(b.2) Generate the new $[X'_{6ik}]_{I\times K}$ denoted $[X^*_{6ik}]_{I\times K}$ =

$$\begin{bmatrix} X^*_{611} = randomize(w'_1) & \cdots & X^*_{61K} = randomize(w'_k) \\ \vdots & & \vdots \\ X^*_{6i1} = randomize(w'_1 - \sum_{q=1}^{i-1} X^*_{6q1}) & \cdots & X^*_{6iK} = randomize(w'_K - \sum_{q=1}^{i-1} X^*_{6qK}) \\ \vdots & & \vdots \\ X^*_{6I1} = w'_1 - \sum_{i=1}^{I-1} X^*_{6i1} & \cdots & X^*_{6IK} = w'_K - \sum_{i=1}^{I-1} X^*_{6ik} \end{bmatrix}$$

***END*.**

5.1 Proposed Heuristic Rules for GA Mutation

It is argued that the main reason why P-GA takes longer time to obtain an acceptable solution in the solving process of fMLN problem can be found in the following part of the algorithm. In the mutation part when one gene of the selected chromosomes is changed randomly, it may bring out the chromosome from the satisfied constraints space and makes infeasible solution. And that is the reason why some heuristic rules are embedded to the mutation so that the chromosome that has been tried can be kept to stay within the feasible region. This also allows the algorithm to speed up to obtain acceptable solution within a reasonable time. The heuristic rules of the embedded mutation of GA by considering the problem constraints are as follows:

Step 1: Let *N*= Maximum $(u_j^D, u_k^R, d_l, \forall j, k, l)$, that is maximum amount of the product (X),

a) Generate $m\_u = Randomize\ (N)$, m_u is the new amount of the gene which it will be mutated.
b) Generate $n\_m = Randomize\ (6)$ +1, n_m denotes the number of type of decision variable (X_1 , $X_2, X_3, X_4, X_5, X_6, y$) that its pertained gene will be mutated.

Step 2: If $n\_m = 1$, find the specific gene of X_1 which it is going to be mutated by the following procedures:

a) Generate $iX_1 = Randomize\ (m)$ that *m* is the number of row of the X_{1ij} matrix that *m=I.*
b) Generate $jX_1 = Randomize\ (n)$ that *n* is the number of column of the X_{1ij} matrix that *n=J.*

Step 3: Let T is the difference between the estimated amount for mutation (m_u) and the current amount of the gene that have been defined in step 2. The new gene is denoted as m_u.

a) $T = X_1(iX_1, jX_1)$ - m_u
b) $X_1(iX_1, jX_1) \longleftarrow m\_u$

Step 4: Let $[X_{2jk}]_{J\times K} = \begin{bmatrix} X_{211} & \cdots & X_{21K} \\ \vdots & \ddots & \vdots \\ X_{2J1} & \cdots & X_{2JK} \end{bmatrix}$ be the matrix of product amount shipped from DC_j to R_k and $[X_{5jl}]_{J\times L} = \begin{bmatrix} X_{511} & \cdots & X_{51L} \\ \vdots & \ddots & \vdots \\ X_{5J1} & \cdots & X_{5JL} \end{bmatrix}$ be the matrix of product amount shipped from DC_j to $Customer_l$.

For j = 1, 2, 3, ⋯, J

a) Add *T* to one of the gene of X_{2jk} or X_{5jl} randomly.
b) For K = 1, 2, 3, ⋯ , *K*
   - If *T* was added to X_{2jk} then subtract *T* from one of the gene of X_{6ik}

END.

It is noted that when n_m = 2 or 3 or 4 or 5 or 6 it has the same process as what has been explained above with the relevant decision variables. For instance if $n\_m = 2$ it is obvious that one of the gene of X_{2jk} will be mutated. This specific gene of X_{2jk} with its changing value (*T)* will be found based on above mentioned procedures, then satisfying those problem constraints which X_{2jk} is involved by adding or subtracting (*T)* to the related decision variables. When n_m = 7, m_u will be generated by binary value {0, 1} and the steps are similar with its relevant decision variables as what has been explained above.

According to above mentioned rules the chances of having negative value are low. In this case the specific decision variable will set to be equal to zero as the following example:

$$\text{if } \sum_{k=1}^{K} X_{6ik} - T < o \text{ , then set } \sum_{k=1}^{K} X_{6ik} = 0.$$

Here the penalty method could be useful for a few chromosomes which are moved to infeasible region, because penalty method was used in HR-GA as it was mentioned earlier.

6. Implementation, Testing and Validation

The data used in this research was generated by the author as the problem case. Diverse 5 problem cases have been created by this research to implement the proposed algorithm and compare the obtained solutions with the standard GA. Hardware platforms employed by the researcher were a 2 GHz processor intel core 2 duo with 1GB memory and running windows 7 professinal. It is noted that the scope of this research did not include establishing nessesary conditions to hardware requirements. The mathematical model of fMLN was translated into program written in Matlab version 7, 2009. The solution obtained of every problem case is the average solution of running 28 times of every proposed algorithm. The problem cases were kept to use for differenet proposed algorithm implementation and the obtained results were displayed in a proper manner in Matlab.

To validate the obtained results, LINDO software was used as well. Although LINDO software has certain restriction when it comes to the number of decision variables, it is however still very useful for very small problem cases with small number of decision variables since LINDO is using simplex method to solve the problems. However to prove and validate the result using LINDO software is useful and necessary to ensure the accuracy of the proposed algorithm.

7. Numerical Experiment

In this section the standard genetic algorithm with penalty methods (P-GA) and HR-GA are applied in five different problem cases with different number of plants (I), DCs (J), retailers (K) and customers (L) besides having different outputs of plants (b_i), different customer demands(d_l), Upper bound of the capacity of DC(UD) and Upper bound of the capacity of retailer (UR). It depends on the number of variables in each problem case where the relevant product quantity of each arc of the network shows the best product delivery route for each customer. The expected result shape looks like the following:

$$[X_{1ij}] = \begin{bmatrix} X_{111} & \cdots & X_{11J} \\ \vdots & \ddots & \vdots \\ X_{1I1} & \cdots & X_{1IJ} \end{bmatrix}, [X_{2jk}] = \begin{bmatrix} X_{211} & \cdots & X_{21K} \\ \vdots & \ddots & \vdots \\ X_{2J1} & \cdots & X_{2JK} \end{bmatrix}, [X_{3kl}] = \begin{bmatrix} X_{311} & \cdots & X_{31L} \\ \vdots & \ddots & \vdots \\ X_{3K1} & \cdots & X_{3KL} \end{bmatrix}$$

$$[X_{4il}] = \begin{bmatrix} X_{411} & \cdots & X_{41L} \\ \vdots & \ddots & \vdots \\ X_{4I1} & \cdots & X_{4IL} \end{bmatrix}, [X_{5jl}] = \begin{bmatrix} X_{511} & \cdots & X_{51L} \\ \vdots & \ddots & \vdots \\ X_{5J1} & \cdots & X_{5JL} \end{bmatrix}, [X_{6ik}] = \begin{bmatrix} X_{611} & \cdots & X_{61K} \\ \vdots & \ddots & \vdots \\ X_{6I1} & \cdots & X_{6IK} \end{bmatrix}$$

$[y_1] = [J] \; \forall j \in \{0,1\}$, $[y_2] = [K] \; \forall k \in \{0,1\}$

where; y_1 , y_2 show which DC and retailer are opened or closed.

It is to observe and show the significance in obtaining better and acceptable results significantly using the heuristics rules embedded GA than conventional GA with penalty methods. The comparison is presented as in the following (Table 1).

Table 1: Solution comparison using HR-GA and P-GA

Problem #	*Number of plants*	*Number of DCs*	*Number of Retailers*	*Number of Customers*	*Number of Decision variables*	*Solution of P-GA*	*Elapsed Time of P-GA (s)*	*Solution of HR-GA*	*Elapsed Time of HR-GA (s)*
1	2	2	2	2	28	3805	90.68	3085	0.26
2	5	9	12	40	1274	32550	305.01	11810	0.63
3	7	11	15	100	3673	92800	816.45	33547	2.76
4	9	13	17	200	8321	224101	4603.20	75132	11.12
5	15	18	22	300	17536	732150	6746.72	181464	39.68

Table 1 show that with the same problem cases the algorithm elapsed time of using HR-GA is more acceptable than the algorithm elapsed time of using P-GA. As it was mentioned before by using heuristic rules embedded GA search is started within the special feasible region where all problem constraints are satisfied besides protecting the chromosomes from coming out of the region. It is obvious that the obtained results of HR-GA are better than the

results of P-GA in terms of solution quality (lower cost) and algorithm elapsed time. The biggest problem case which is problem case number 5 has 17536 numbers of decision variables and it is such a large problem case that HR-GA could solve it efficiently as well as other problem cases.

8. Conclusion

In this paper the multi source single product flexible multistage logistics network problem was considered. According to the solution algorithms used by [9], customer order was not allowed to be split and to be fulfilled from different facilities at the same time. This research has attempted to overcome this problem by allowing that every customer can be served by multi facilities. That is, the customer order is allowed to be split.

To meet the mentioned objective, one algorithm has been proposed and developed to solve the multi source single product fMLN problem. The proposed algorithm is GA with Heuristics Rules for P-GA (HR-GA).

Based on the experimental results it is proven that Although P-GA could solve the multi source single product fMLN, however the elapsed time of the algorithm is not reasonable. To overcome this problem new heuristics rules were proposed that can be embedded in GA to obtain an acceptable solution within reasonable time. The proposed algorithm is called HR-GA. The result of using HR-GA was extremely better than the result of using P-GA in terms of the acceptable solution and elapsed time. The result of P-GA and HR-GA was proven and verified using LINDO software as well.

Additionally in this research the new chromosome representation that was used for P-GA and HR-GA showed that every gene of the chromosome is one of the decision variables in comparison with the previous papers in solving multi source single product fMLN problem.

Acknowledgment

The authors sincerely thank to Universiti Technologi Malaysia and Ministry of Higher Education (MOHE) Malaysia for sponsoring this research under FRGS (Vot. 78502).

REFERENCES

1. Alim, F., Ivanov, K. (2005). *Heuristic Rules Embedded Genetic Algorithm to Solve In-Core Fuel Management Optimization Problem, GECCO'05*, Washington, DC, USA.
2. Altiparmak, F., Gen, M., Lin, L., Paksoy, T. (2006). *A genetic algorithm approach for multi-objective optimization of supply chain networks*.
3. Celik, H.M. (2004). *Modeling freight distribution using artificial neural networks*, Journal of Transport Geography 12 (2) 141–148.
4. Chalewicz, Z., Vignaux, G. A. and Hobbs, M. (1991). A non-standard genetic algorithm for the nonlinear transportation problem, *ORSA Journal on Computing*, 3(4), 307–316.
5. Costa, A., Celano, G., Fichera, S., Trovato, T. (2010). *A new efficient encoding/decoding procedure for the design of a supply chain network with genetic algorithms*, Computers & Industrial Engineering 59 986–999.
6. Craenen, B.G.W., Eiben, A.E. Marchiori, E. (2002). *Solving Constraint Satisfaction Problems with Heuristic-based Evolutionary Algorithms* - ieeexplore.ieee.org.
7. Dullaert .W., Maes .B., Vernimmen,.B. Witlox .F. (2005). *An evolutionary algorithm for order splitting with multiple transport alternatives, Expert Systems with Applications* 28 (2) 201–208.
8. Gen, M., Altiparmak, F., Lin, L. (2006). *A genetic algorithm for two-stage transportation problem using priority-based encoding*.
9. Gen, M., Cheng. R., Lin, L. (2008). *Network Models and Optimization, Multiobjective Genetic Algorithm Approach*, Springer-Verlag London Limited.
10. Gen, M., Syarif, A. (2005). *Hybrid genetic algorithm for multi-time period production/distribution planning*.

11. Jayaraman, V. & Pirkul, H. (2001). *Planning and coordination of production and distribution facilities for multiple commodities*, European Journal of Operational Research, 133, 394– 408.

12. Lee, J. E., Rhee, K. G. & Gen, M. (2007). *Designing a reverse logistics network by priority based genetic algorithm*, Proceeding of International Conference on Intelligent Manufacturing Logistics System, 158–163.

13. Li, Y. Z., Gen, M. & Ida, K. (1998). *Improved genetic algorithm for solving multi-objective solid transportation problem with fuzzy number*, Japanese Journal of Fuzzy Theory and Systems, 4(3), 220–229.

14. Lin, L., Gen, M. and Wang, X. (2007). *A Hybrid Genetic Algorithm for Logistics Network Design with Flexible Multistage Model.*

15. Lin, L., Gen, M., Wang X. (2008). *Integrated multistage logistics network design by using hybrid evolutionary algorithm*.

16. Paksoy, T., Özceylan, E., WEBER, G.W. (2009). *A Multi- objective Mixed Integer Programming Model For Multi Echelon Supply Chain Network Design and Optimization.*

17. Sheta, A., Turabieh, S. (2006). *A Comparison between Genetic Algorithms and Sequential Quadratic Programming in Solving Constrained Optimization Problems*, AIML Journal, Volume (6), Issue (1), January.

18. Shimizu, Y., Matsuda S. and Wada, T. (2006). *A Flexible Design of Logistic Network against Uncertain Demands through Hybrid Meta-heuristic Method.*

19. Shimizu, Y., & Hiroshi, K. (2008). *An Implementation of Parallel Computing for Hierarchical Logistic Network Design Optimization Using PSO*.

20. Syarif, A. & Gen, M. (2003). *Double spanning tree-based genetic algorithm for two stage transportation problem*, International Journal of Knowledge-Based Intelligent Engineering System. Systems, 8, 70–78.

21. Yaohua, H., Hui, C.W. (2007). *Genetic algorithm based on heuristic rules for high-constrained large-size single-stage multi-product scheduling with parallel units.*

22. Yeniay, O. (2005). *Penallty function methods for constrained optimization with genetic algorithms*, Mathematical and Computational Applications, Vol. 10, No. 1, pp. 45-56.