IEMS Vol. 8, No. 4, pp. 228-238, December 2009.

An Agent Gaming and Genetic Algorithm Hybrid Method for Factory Location Setting and Factory/Supplier Selection Problems

Feng-Cheng Yang †
Graduate Institute of Industrial Engineering, National Taiwan University
1, Sec. 4, Roosevelt Rd., Taipei City 10617, TAIWAN
Tel: +886-2-33669503, E-mail: iefcyang@ntu.edu.tw

Shih-Lin Kao
Graduate Institute of Industrial Engineering, National Taiwan University
1, Sec. 4, Roosevelt Rd., Taipei City 10617, TAIWAN
Tel: +886-2-33669503

Received, March 31, 2009; Revised, July 15, 2009; Accepted, August 17, 2009

Abstract. This paper first presents two supply chain design problems: 1) a factory location setting and factory selection problem, and 2) a factory location setting and factory/supplier selection problem. The first involves a number of location known retailers choosing one factory to supply their demands from a number of factories whose locations are to be determined. The goal is to minimize the transportation and manufacturing cost to satisfy the demands. The problem is then augmented into the second problem, where the procurement cost of the raw materials from a chosen material supplier (from a number of suppliers) is considered for each factory. Economic beneficial is taken into account in the cost evaluation. Therefore, the partner selections will influence the cost of the supply chain significantly. To solve these problems, an agent gaming and genetic algorithm hybrid method (AGGAHM) is proposed. The AGGAHM consecutively and alternatively enable and disable the advancement of agent gaming and the evolution of genetic computation. Computation results on solving a number of examples by the AGGAHM were compared with those from methods of a general genetic algorithm and a mutual frozen genetic algorithm. Results showed that the AGGAHM outperforms the methods solely using genetic algorithms. In addition, various parameter settings are tested and discussed to facilitate the supply chain designs.

Keywords: Supply Chain Design, Agent Gaming, Two-stage Genetic Operation, Genetic Algorithm

1. INTRODUCTION

With the increasing speed of the market globalization in the last two decades, supply chain design has become a challenging issue in supply chain management. In general, a supply chain composes six business entities and encompasses two integrated business processes (Beamon, 1998), as illustrated in Figure 1.

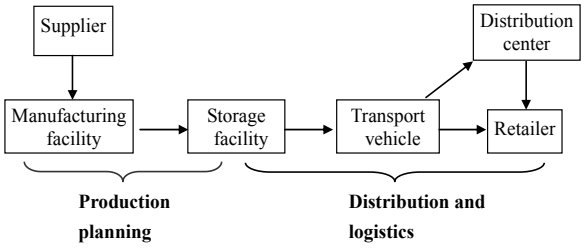

Figure 1. A typical supply chain (Beamon, 1998).

Due to the increasing number of entities involved in a supply chain and much more complicated business processes are conducted, the selections of decision variables and performance indices become important issues in designing a supply chain (Chen *et al*., 2008). Since the transportation cost plays an important role in the performance of a supply chain design (Cohen and Moon, 1990), Chopra and Meind (2004) pointed out that the number and locations of the factories and the selections of their material suppliers should be considered impor-

† : Corresponding Author

tant decision variables in supply chain designs. Song (2006) considered a supply chain network dealing with multi-period production, inventory and distribution planning as a dynamic assignment model. Then, a solution algorithm was developed based on the Benders' decomposition approach. Numerical results showed the method can solve the complex problem effectively.

Prasertwattana and Chiadamrong (2004) analyzed supply chains consisting of one manufacturer and multiple retailers and used GA to determine the appropriate ordering and inventory level to maximize the profit of the chain. Mexixell and Gargeya (2005) used categories of factory capacity, product source, and material source to classify the models of global supply chains. In general, the minimum cost and maximum profit are the most popular performance indices adopted in supply chain modeling (Mexixell and Gargeya, 2005).

While previous researchers had worked on solutions to various design problems of a supply chain, the settings of factory locations in a two-dimension region were seldom discussed. This paper proposes an effective method to model an entity location setting and selection problem to supplement the discussion of the partner selection problem in a supply chain. In particular, the location setting of the entities is landed on a two-dimensional and continuous coordinate space. Those locations of factories (x and y coordinates) and the connections between the selected factory and the retailer are the decision variables, and the objective function is to minimize the total cost.

An agent-based economics model is an adaptive system that consists of a collection of autonomous agents interacting in various rules to achieve their own interests. With the increasing computation speed of computers, agent-based systems have been successfully applied to market related application fields, such as stock markets, foreign exchange markets, and labor markets (Tesfatsion, 2001). Although the model mainly concerns the objective of each individual agent, recent research has shown that it also has well performance in solving optimization problems of discrete variables. For example, Ishibuchi *et al*. (2001) adopted an agent-based computational economics model to develop an evolved, unplanned coordination method to evolve the result of a market selection game, where a repeated game played with many agents and several markets. In the game, each agent has to choose a market with a high selling price to sell its products to maximize its profit. The market price is determined by the amount of products supplied to the market. More products aggregating to a market will yield a lower selling price. For comparison, they implemented a genetic algorithm to evolve this market selection problem to maximize the summation of all agents' profits. Results showed that their method obtained similar results with that of the genetic algorithm. However, the computation time of their coordination method was significantly less than that of the GA.

To solve the factory location setting and factory/supplier selection problem, which contains both continuous and discrete variables, a method combining the agent-based gaming and GA operations is proposed in this paper. An interlocking mechanism is applied to these two computation modules during the evolution procedure. The mechanism was initiated by Ong and Tan (2002) for solving the placement planning problem for high-speed printed circuit board assembly machines. In their problem, the feeder arrangement and the component placement sequencing are two different kinds of variables, yet they constitute a complete solution. Therefore, their method engaged a two-stage genetic operation to evolve these two kinds of variables alternatively. At a time, only a kind of variables is under evolution, while holding the other. In our model, the location variables of the factories and the factory/supplier selection variables are comparably two different kinds of variables. They will be alternatively evolved by GA operations and repeated agent gaming operations, respectively.

2. PROBLEM FORMULATION

In this paper, we will introduce a two-tier supply chain decision model first and then a three-tier one subsequently.

2.1 Factory Location Setting and Selection Problem

The first two-tier supply chain model yields the first factory location setting and selection problem (FLSSP). Figure 2 is an example of the FLSSP. The numbers adjacent to the retailers are their product demands; conversely those adjacent to the factories are their production quantities. In addition to these known information, the locations (x and y coordinates) of the retailers are known, while the locations of the factories are to be determined. In this model, a company plans to construct factories on the rectangular area, the number of factories and locations are to be determined; however, a maximal number of factories is planned. After that each retailer will select a nearest factory to supply his product demands. When each retailer selects its supplying factory, the production unit cost and production quantity of each factory are determined by the economics of scale rule. In addition, the transportation cost is

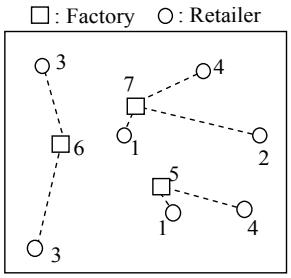

Figure 2. An example of the FLSSP.

taken into account in computing the overall cost of a supply chain design. Therefore, minimizing the sum of the production and transportation costs is naturally the practical goal for this supply chain design problem.

The following assumptions are adopted in this model:
1. The product demand of each retailer is known.
2. Each retailer selects only one factory to supply its demands.
3. Factory locations are set within the defined rectangular region.
4. Production costs of all factories share the same economics of scale rule.
5. The production capacity of each factory can satisfy the demands of all its downstream retailers.

Let the region be a rectangular area in a two-dimensional space, with x coordinate bounded by $l^{(x)} \le x \le u^{(x)}$ and coordinate y by $l^{(y)} \le y \le u^{(y)}$. The transportation cost of one shipment from factory i to retailer j is

$$t^r_{i,j} = T^r \sqrt{\left(x^f_i - x^r_j\right)^2 + \left(y^f_i - y^r_j\right)^2}\,, \tag{1}$$

where T^r is the transportation cost per unit distance, x^f_i and y^f_i are the x and y coordinates of factory i; and x^r_j and y^r_j are coordinates of retailer j. Each retailer can select only one factory to supply its demands cal be modeled as a constraint equation,

$$\sum_{i=1}^{l} z^r_{i,j} = 1\,,\ \forall j \in \{1, 2, \cdots, h\};\ \ z^r_{i,j} \in \{0, 1\}\,, \tag{2}$$

where $z^r_{i,j}$ is a binary decision variable for factory i and retailer j, l is the number of factories, and h is the number of retailers. When factory i supplies products to retailer j, the value of $z^r_{i,j}$ is set to 1. Otherwise, the value is 0. After each retailer selects its factory, the production quantity of each factory can be obtained by summing up the demands from its retailers. Thus, the production quantity of factory i is

$$u^f_i = \sum_{j=1}^{h} \left(z^r_{i,j} \cdot q_j\right), \tag{3}$$

where q_j is the quantity of product demand of retailer j. The frequently used power cost function (Luss, 1982) is used to calculate the total production cost. The general form of the function is

$$f(u) = Gu^{\alpha},\ 0 < \alpha < 1,\ G > 0\,, \tag{4}$$

where G is a positive number, α is a real number between 0 and 1, and u is the production quantity. Hence, the production cost per unit can be calculated by dividing the total production cost with the total production quantity. The production cost for one product of factory i is

$$c_i = \frac{G \cdot \left(u^f_i\right)^{\alpha}}{u^f_i} = G \cdot \left(u^f_i\right)^{\alpha - 1}. \tag{5}$$

The objective of this problem is to minimize the total production and transportation cost of the model. For real applications, there is a transportation batch limit b applied to our model. Once the quantity of products exceeds b, another shipment is required. Therefore, the objective of the FLSFSSP is

$$\text{Min } Z_1 = \sum_{j=1}^{h}\sum_{i=1}^{l}\left(z^r_{i,j} \cdot c_i \cdot q_j\right) + \sum_{j=1}^{h}\sum_{i=1}^{l}\left(z^r_{i,j} \cdot t^r_{i,j}\left\lceil \frac{q_j}{b} \right\rceil\right). \tag{6}$$

2.2 Factory Location Setting and Factory/Supplier Selection Problem

The second model presented is a three-tier supply chain decision model, which yields a factory location setting and factory/supplier selection problem (FLSF SSP). Figure 3 shows that the FLSFSSP consists of material suppliers, manufacturing factories, and retailers. In addition to setting locations of factories and selecting factories for retailers, each factory must select a supplier to procure the raw materials for product production. Due to the economics of scale, the more materials a supplier sells, the cheaper the material price. Therefore, the raw material procurement cost and the transportation cost are considered in selecting a raw material supplier for each factory. For simplicity, one unit of raw material is required to produce one unit of final product.

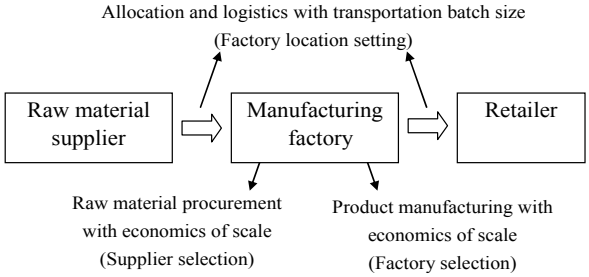

Figure 3. The three-tier supply chain structure of the FLSFSSP.

Figure 4 shows an example of the FLSFSSP. The numbers near the retailers are their product demands, near the factories are their production quantities, and near the suppliers are the amount of raw material provided by them. Within the defined two-dimension region, the numbers and locations of retailers and suppliers are all known and the product demands of each retailer are given. A company is to construct a number of factories on this region to establish the three-tier supply chain. Decision variables include (1) the locations of the factories, (2) the partnerships between suppliers and factories, and (3) the partnerships between factories and retailers. When the decision variables are set, the production quantity and production cost of each factory and

the sale quantity and selling price of the raw material of each supplier are computed based on the economics of scale rule. As expected, the objective of the FLSFSSP is to minimize the total material procurement and transportation cost and the production and transportation cost of the products.

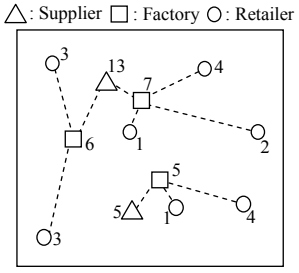

Figure 4. An example of the FLSFSSP.

In addition to the assumptions described in Section 2.1, two additional ones are applied:

1. Each factory can select only one supplier to purchase raw materials.
2. The supplier can provide the required raw material without any limitation and one unit of raw material is required for producing one unit of product.

Similar to Eq. 1, the transportation cost of one shipment of raw material from supplier k to factory i is

$$t_{i,k}^{f} = T^{f} \sqrt{\left(x_i^f - x_k^s\right)^2 + \left(y_i^f - y_k^s\right)^2}\,, \tag{7}$$

where T^f is the transportation cost per unit distance, x_k^s and y_k^s are the x and y coordinates of supplier k. Each factory can select only one supplier to purchase raw materials is a constraint defined by

$$\sum_{k=1}^{n} z_{i,k}^{f} = 1,\ z_{i,k}^{f} \in \{0, 1\}\,,\ \forall i \in \{1, 2, \cdots, l\}\,, \tag{8}$$

where $z_{i,k}^{f}$ is a binary decision variables for associating supplier k with factory i, n is the number of suppliers under discussion. If factory i purchases raw materials from supplier k, the value of $z_{i,k}^{f}$ is set to 1. Otherwise, the value is 0. The production quantity of each factory is defined in Eq. 3. The total quantity of raw materials provided by supplier k is obtained by summing up the production quantity of those factories associated with supplier k as

$$u_k^s = \sum_{i=1}^{l} \left(z_{i,k}^{f} \cdot u_i^f\right). \tag{9}$$

Similar to the definition of production cost, the raw material price of supplier k is also calculated by the power cost function shown in Eq. 4 as

$$e_k = \frac{B \cdot \left(u_k^s\right)^{\beta}}{u_k^s} = B \cdot \left(u_k^s\right)^{\beta - 1}, \tag{10}$$

where B is a positive number and β is a real number between 0 and 1.

The objective of the is then to minimize the sum of the total production cost, raw material procurement cost, and transportation costs of products and materials. Without lost of generality, the same transportation batch limit b applied to products is applied to raw materials too. The total raw material procurement cost of a factory is evaluated by multiplying the quantity needed with the raw material selling price of the selected supplier. Therefore, the objective of the FLSFSSP is

$$\begin{aligned} \text{Min } Z_2 = &\sum_{j=1}^{h}\sum_{i=1}^{l}\left(z_{i,j}^{r} \cdot c_i \cdot q_j\right) + \sum_{i=1}^{l}\sum_{k=1}^{n}\left(z_{i,k}^{f} \cdot e_k \cdot u_i^f\right) \\ &+ \sum_{j=1}^{h}\sum_{i=1}^{l}\left(z_{i,j}^{r} \cdot t_{i,j}^{r}\left\lceil \frac{q_j}{b} \right\rceil\right) + \sum_{i=1}^{l}\sum_{k=1}^{n}\left(z_{i,k}^{f} \cdot t_{i,k}^{f}\left\lceil \frac{u_i^f}{b} \right\rceil\right). \end{aligned} \tag{11}$$

3. REPEATED AGENT GAMING AND GENETIC ALGORITHM HYBRID METHOD

Notice that, each presented problem has two kinds of decision variables. In the FLSSP, the factory selections for the retailers are discrete variables and the locations of factories are continuous variables. In the FLSFSSP, the factory selections for the retailers and the supplier selections for the factories are discrete variables; while the locations of factories are continuous variables. In this paper, an agent gaming and genetic algorithm hybrid method (AGGAHM) is proposed to solve these problems. Figure 5 is the framework of the AGGAHM. The repeated agent gaming computation is to deal with selection variables and the genetic computation is for the factory locations. Both computations are alternatively executed to evolve the discrete and continuous variables separately.

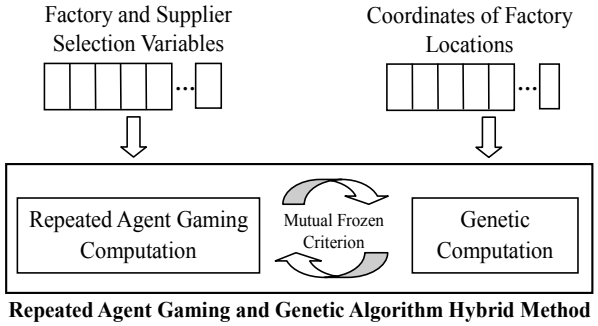

Figure 5. The framework of the AGGAHM.

When the agent gaming computation is executing, the evolution of the coordinate variables is frozen, until the frozen condition is broken. Conversely, the agent gaming for the selection variables is alternatively frozen and the genetic computation starts execution to evolve the factory locations. Once the frozen condition is broken again, the agent gaming computation takes charge

over the solution evolution. These computations are exclusively executed until a stop criterion is met.

3.1 Repeated Agent Gaming Computation

In the agent gaming computation, the supply chain entity that needs decision is regarded as an agent. In each round of the gaming, all the agents make their decisions to select one supply chain partner. The decision making of the agent is referring to its best decision experience or to one of its neighboring agents. Note that the next decision might lead to a worsen result than the current one. In the FLSSP, each retailer is modeled as an agent to select a factory in the gaming. In the FLSFSSP, each factory is modeled as an agent to select a raw material supplier.

During the agent gaming computation, the evolution of the variables of factory locations is frozen. The current best coordinates of factory locations are used to associate with the selections of agents to evaluate their solutions.

Before the agent gaming computation starts, each agent will identify its neighboring agents from the locations of the agents. At first the selections of the agents and the locations of the factories are initialized randomly. The values of them are considered the current best solution initially. During the gaming, a selection replacement operation and a selection mutation operation are subsequently executed to make decisions for each round of the gaming. In each round, the objective value is evaluated to update the current best solution, if a better solution is resulted. The agent gaming procedure is repeated until reaching one stopping criterion or the frozen condition is broken. The details of the repeated agent gaming computation for the two problems are described in the next two sections.

3.1.1 Repeated Agent Gaming Computation for the FLSSP

In solving the FLSSP, an agent stands for a retailer to select a factory to satisfy its demands. Before the AGGAHM method starts, two operations are executed first:

(1) Initialize the factory selections of the agents and the locations of the factories randomly and set the initial solution the best solution so far.
(2) Construct the set of neighboring agents for each agent.

A neighborhood size q is given by the user to set the set of neighboring agents, where $2 \le q \le h$. Note that h is the number of retailers and there should be at least two agents to form a neighbor set. Therefore, for agent j, the $q-1$ nearest agents are to be identified as its neighbors to form the neighbor set N_j. Since neighbors are close to each other, so do the transportation costs for their demands. The factory selections of its neighbors will be referred by the agent to make the next decision in the gaming. The gaming for an agent is to refer the decisions of its neighbors and its own experience to make the next decision.

The procedure of constructing neighbor sets is given in Figure 6.

Input:
h: Number of retailers
q: Number of neighbors
x_j^r: X-coordinate of retailer j, $\forall j = 1, 2, ..., h$
y_j^r: Y-coordinate of retailer j, $\forall j = 1, 2, ..., h$
$D_{j,g}$: Distance between retailer j and retailer g, $\forall j, g = 1, 2, ..., h$
Output:
$e_{j,p}$: Retailer index of retailer j's p th neighbor
$\forall j = 1, 2, ..., h$, $p = 1, 2, ..., q$
$\boldsymbol{N}_j$: Set of the retailer indexes of retailer j's neighbors

Procedure BuildUpTheNeighborsOfEachRetailer
begin
  for $j = 1$ **to** h **do**
    $\boldsymbol{A} \leftarrow \{1, 2, ..., h\}$
    $\boldsymbol{N}_j \leftarrow \{\ \}$
    for $p = 1$ **to** q **do**
      $e_{j,p} \leftarrow \arg \min_{\forall g \in \boldsymbol{A}} \{D_{j,g}\}$
      $\boldsymbol{A} \leftarrow \boldsymbol{A} - \{e_{j,p}\}$
      $\boldsymbol{N}_j \leftarrow \boldsymbol{N}_j \cup \{e_{j,p}\}$
    end for
  end for
end

Figure 6. The procedure of constructing neighbor sets.

The primary operations for an agent in the gaming are selection replacement and selection mutation.

● Selection Replacement Operation

Each agent investigates the factory selections of its neighbors and its own best selection experience to make the next selection. Before the method starts, values of a selection replacement rate P^r and a selection imitation rate P^a are specified by the user, where $0 < P^r, P^a < 1.0$.

The selection replacement operation is stochastically executed. A random number $\theta^r \sim \mathrm{U}(0,1)$ is generated first. The selection replacement operations executes only if $\theta^r < P^r$. In the operation, another random number $\theta^a \sim \mathrm{U}(0,1)$ is generated. If $\theta^a < P^a$, the agent adopts one of its neighbors' selections as the new selection for the next round. Otherwise, the agent takes its own best factory selection as the next round selection.

The transportation costs from factories to retailers and the production costs of the products are changed when the retailers (agents) change their factory selections. Therefore, the cost of retailer j's factory selection in the tth gaming round is

$$V_{j,t} = t^r_{S_{j,t},j} + c_{S_{j,t}}, \tag{12}$$

where $s_{j,t}$ is the factory selected by retailer j in the tth round, $t^r_{S_{j,t},j}$ is the transportation cost from factory $s_{j,t}$ to retailer j, and $c_{S_{j,t}}$ is the production cost per unit of the factory. The result of the cost evaluation is then used

to optionally update the best selection of the agent and to identify which neighboring agent has the lowest cost. When an agent is computationally decided to imitate the selection of a neighbor, the neighbor is stochastically chosen based on its rightness of the factory selection. The rightness is quantified as a selection probability for the stochastic selection. Therefore, a neighbor with a lower cost is assigned with a larger probability. To guarantee a positive value of probability is assigned to the worst neighbor that has the highest cost, a *worst cost threshold* is defined from the costs obtained from the set of neighboring agents. The threshold for agent *j* is

$$\hat{V}_j = 2 \max_{\forall a \in N_j} \{V_{a,t}\} - \frac{\sum_{\forall a \in N_j} V_{a,t}}{|N_j|}. \quad (13)$$

The imitation probability of a neighboring agent *p* in N_j for agent *j* at the *t*th round is

$$P_{j,p,t} = \frac{\hat{V}_j - V_{p,t}}{\sum_{\forall a \in N_j} (\hat{V}_j - V_{a,t})}, \forall p \in N_j, \quad (14)$$

$$\sum_{p \in N_j} P_{j,p,t} = 1. \quad (15)$$

In the imitating operation for agent *j*, a uniformly distributed random number is generated to stochastically select one neighbor, say agent p^*, from the pool of the neighboring agents, N_j. Then the factory selection for agent *j* at the (*t*+1)th round is replaced with the selection of agent p^* at the *t*th round; i.e.,

$$S_{j,t+1} \leftarrow S_{p^*,t}.$$

● Selection Mutation Operation

Before the method starts, the user have to set a selection mutation rate P^m, a real number between 0 and 1. The mutation operation on each agent generates a random number $\theta^m \sim U(0,1)$ first. If $\theta^m < P^m$, the agent discards the current factory selection and change it to a randomly selected factory.

3.1.2 Repeated Agent Gaming Computation for the FLSFSSP

In the FLSFSSP, the retailers and factories are all modeled as agents to select their factories and suppliers in each gaming round. The agent gaming computation for the factory selection is the same as that executed in solving the FLSSP. However the cost of retailer *j* is

$$V'_{j,t} = t^r_{S_{j,t},j} + c_{S_{j,t}} + e_{B_{S_{j,t},t}}, \quad (16)$$

where $B_{S_{j,t},t}$ is the selected supplier of the selected factory $s_{j,t}$ and $e_{B_{S_{j,t},t}}$ is the raw material cost. Moreover, the gaming operations of the supplier selection for the factories are similar to the factory selections for the retailers. In particular, the formation of the neighboring factory set R_i for each factory *i* is dynamically changed, since there locations are the decision variables. Therefore, in each agent gaming round, the neighbor sets of all the factories should be reconstructed first.

Similarly, a neighborhood sized $2 \le f \le l$ is given by the user to construct the neighbor set R_i for each factory *i*. The formation procedure is similar to that of the retailer set, which was shown in Fig. 6. However, the distances between factories are changed in each round of the agent gaming. The selection replacement operations for retailer agents in the FLSFSSP are exactly the same as in the FLSSP. Similar steps are executed for each factory agent to make its next selection of raw material supplier.

When the run-time generated random number passes the thresholds of the replacement rate P^r and imitation rate P^a, the supplier selection of a factory is changed to the selection of one of its neighbors. The neighbor will be probability proportionally chosen from the neighbor set based on their "selection rightness." Similarly, the rightness is evaluated from the transportation cost and the raw material selling price incurred from the supplier selection. The sum of the incurred cost from the supplier selection of factory *i* at the *t*th round is

$$K_{i,t} = t^f_{i,B_{i,t}} + e_{B_{i,t}}, \quad (17)$$

where $t^f_{i,B_{i,t}}$ is the transportation cost per shipment from the selected supplier $B_{i,t}$ to factory *i* and $e_{B_{i,t}}$ is the material selling price of the supplier.

Similar techniques are adopted in defining the selection probability for the supplier selection imitation. In order to have a positive probability for each neighboring factory, a worst threshold of the cost is defined as

$$\hat{K}_i = 2 \max_{\forall w \in R_i} \{K_{w,t}\} - \frac{\sum_{\forall w \in R_i} K_{w,t}}{|R_i|}, \quad (18)$$

where R_i is the neighbor set of factory *i*. Then, The probability for neighbor factory *p* to be chosen for the imitation of factory *i* is

$$F_{i,p,t} = \frac{\hat{K}_i - K_{p,t}}{\sum_{\forall w \in R_i} (\hat{K}_i - K_{w,t})}, \forall p \in R_i, \quad (19)$$

$$\sum_{\forall p \in R_i} F_{i,p,t} = 1. \quad (20)$$

After calculating the probabilities of Eq. 19, a uniformly distributed random number is generated to stochastically select one factory, say factory p^*, from the neighbor set R_i. Then the supplier selection for factory *i* at the (*t*+1)th round is replaced with the selection of factory p^* at the *t*th round; i.e.,

$$B_{i,t+1} \leftarrow B_{p^*,t}.$$

As discussed, the mutation operation is stochastically executed, where the supplier selection of a factory is replaced with a randomly selected supplier.

3.1.3 Overall Process of the Repeated Agent Gaming

On the basis of the operations described above, the steps of the repeated agent gaming computation are:

Step 1: Initialize the agent selections randomly. Assign each initial selection as the agents' best selection. The initial selections are also stored with the current best coordinates as the current best solution.

Step 2: Construct the neighbor sets of retailers for each retailer.

Step 3: Construct the neighbor sets of factories for each factory.

Step 4: Execute the selection replacement operation to make new selection for each agent.

Step 5: Execute the selection mutation operation to update the selection for reach agent.

Step 6: Calculate the resulting cost values for all agents. Update the current best selection, if the overall cost value is smaller than the current best solution. Update the current best selection for each agent, if the selection is better than its best selection.

Step 7: Stop the agent gaming computation if one stopping criterion is met or the frozen condition of genetic algorithm is broken. Otherwise, go back to Step 3.

3.2 Genetic Computation

The Genetic Algorithm proposed by Holland (1975) is one of the most popular optimization algorithms and has been applied in various fields to solve different kinds of decision making problems. The decision variables are encoded as genes of a chromosome and a chromosome usually represents a solution to the problem. After the settings of population size, crossover and mutation rates, the population of chromosomes is constructed and initialized with initial values (usually random values). Then the chromosomes are subject to a serial of evolution operations: reproduction, crossover, and mutation operations, to evolve the solutions to the optimal one

The Evolver API (2001) published by Palisade Inc. is a GA programming library, which contains a set of genetic operations and provides six encoding modes for modeling different kinds of decision variables. This paper used the Evolver API to implement the genetic computation module to evolve the coordinates of factory locations. The coordinates of the factories are encoded by the "Recipe" mode, as hown in Figure 7.

x_1^f	y_1^f	x_2^f	y_2^f	• • •	x_l^f	y_l^f

Figure 7. Chromosome encoded with the coordinates of factory locations.

The computation flow of the genetic computation for the FLSSP and FLSFSSP is the same. When the genetic computation is executing, the gaming of the selection variables is frozen. The current best selections are used to associate with the coordinates evolved by the genetic computation to evaluate the objective value of the current solution.

During the evolution computation, the current best objective value and the current best coordinates of factory locations are updated whenever a better solution is resulted. The genetic computation stops when a stopping condition is met or the frozen condition is broken.

3.3 Procedure Overview of the AGGAHM

Figure 8 shows the computation flow of the proposed AGGAHM. At the beginning, an initial solution is generated randomly and is stored as the current best solution. The solution consists of the current best selections and best coordinates of factory locations. The genetic computation is frozen and the current best coordinates are used for the selection agent gaming to evaluate the objective value of each round of the gaming. The number of gaming rounds can be set as the frozen condition of the genetic computation. Once the number of rounds reaches the limit, the frozen condition is broken and the genetic computation starts evolving the coordinates and the agent gaming computation is frozen. The gaming computation and genetic computation are mutual exclusively and alternatively executed until a stopping criterion is met.

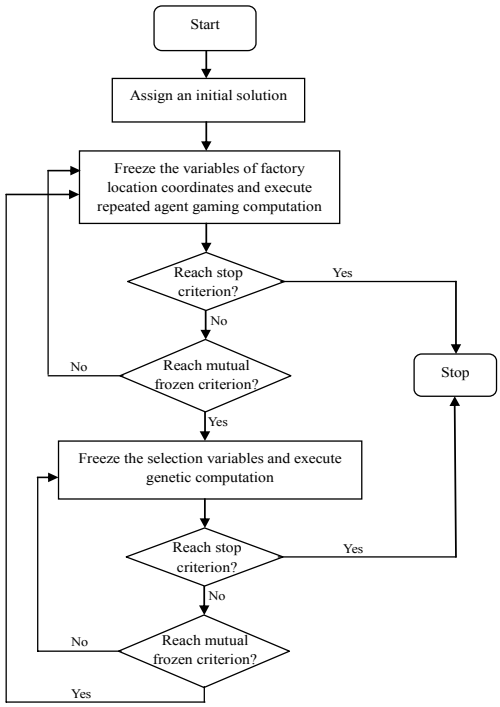

Figure 8. The computation flow overview of the AGGAHM.

4. NUMERICAL TESTS AND RESULT COMPARISON

In this section, the robustness of the AGGAHM is verified by comparing results with a method using general genetic algorithms (GA) and a GA method with a mutual frozen mechanism (MFGA), for solving both of the FLSSP and FLSFSSP. After that three FLSSP examples with different economic scales are tested to discuss the effect of economics scale in supply chain designs. Finally the effect of the frozen conditions on the obtained solutions is studied and discussed.

4.1 Robustness of the AGGAHM

The method with typical genetic algorithm (GA) and the method with mutual frozen genetic algorithm (MFGA) were both implemented by the Evolver API. Note that the MFGA is implemented based on the two-stage genetic program proposed by Ong and Tan (2002). Since no agent gaming computation is involved, two chromosomes separately representing the selection variables and the coordinates of factories are modeled in these methods. The selection variables are encoded by the "Grouping" mode while the coordinates are encoded by the "Recipe" mode in the Evolver. In the GA method, these two chromosomes are evolved simultaneously. On the other hand, the selection and coordinate variables are separately and alternatively evolved in the MFGA method.

The frozen conditions of the MFGA and our AGGAHM are both set as the continuous number of iterations without solution improvement on the current best solution does not exceed a limit. The stopping criteria for the three methods are all set as reaching a maximal number of iterations, where an iteration is a round of gaming or a genetic generation.

4.1.1 Performance of the AGGAHM on the FLSSP

Four examples of the FLSSP were tested to verify the performance of the AGGAHM. Two examples are generated randomly and the others are constructed with a known optimum solution. Table 1 lists the primary data of these examples. In cases 1 and 2, each retailer's demand is randomly set to 1 to 4, while all retailers' demand is set to 3 in cases 3 and 4.

Table 1. The primary features of the FLSSP examples.

Case	Number of Retailers	Maximal Number of Factories	The Optimal Value
Case 1	100	4	Unknown
Case 2	200	8	Unknown
Case 3	100	4	26592.6
Case 4	100	2	25526.6

Figure 9 shows the distribution of the retailers for the four cases, where each dot represents a retailer. The range of the x and y coordinates is from 0 to 100 for all cases. In case 3, only 4 locations are defined with each location situated with 25 retailers. In case 4, there are 50 retailers located at each of the two locations.

The parameter settings for the AGGAHM are (1) the size of neighbors for the retailer and factory is set to 6; (2) the selection replacement rate, imitation rate, and selection mutation rate are 0.9, 0.8, and 0.05 respectively; (3) the population size is 50; (4) the crossover rate and mutation rate of the genetic computation are 0.9 and 0.1; (5) the frozen condition is 60 iterations without improvement; and (6) the stop criterion is 60,000 iterations. Comparatively, the population sizes, crossover rates, mutation rates, and stopping criterions of the GA and MFGA methods are the same as (3), (4) and (6). The mutual frozen condition of the MFGA is to 60 iterations as well. The resulting objective values of the three methods are shown in Table 2.

The objective values obtained from the AGGAHM for the four examples are all the best ones among the three methods. Moreover, notice that in cases 3 and 4, the best objective values found by the AGGAHM are very close to the optimal values listed in Table 1.

Figure 10 illustrates the variation of the current best objective values obtained from the three methods. Notably, the performance of the AGGAHM outperforms others' and the objective value converges toward the near optimal solution quicker than others.

4.1.2 Performance of the AGGAHM on the FLSFSSP

Three examples, as shown in Table 3, were created to test the performance of AGGAHM on the FLSFSSP. Cases 1 and 2 are generated randomly, while the optimal objective value of case 3 is known. In cases 1 and 2, each retailer's demand is randomly set to a value between 1 and 4, while in case 3 the demand of each retailer is 3 products.

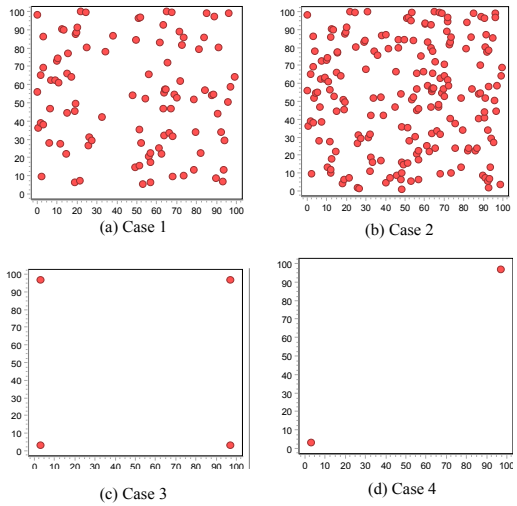

Figure 9. The location distributions of retailers of the four examples of the FLSSP.

Table 2. Objective values obtained from the three methods.

Case	GA	MFGA	AGGAHM
Case 1	28782.39	28415.83	28287.58
Case 2	58256.82	59184.02	52840.14
Case 3	27688.08	28011.56	26882.26
Case 4	26183.59	25966.21	25529.27

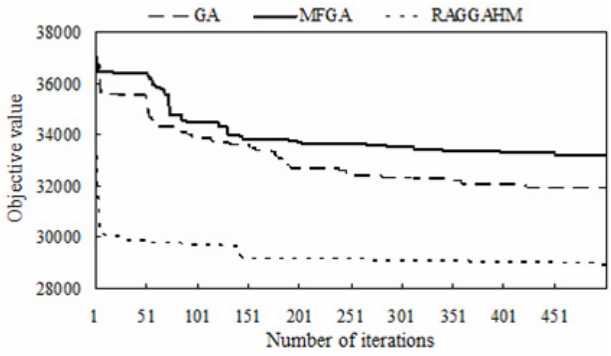

Figure 10. The performance of the three methods in solving the FLSSP.

Figure 11 shows the distribution of the retailers and raw material suppliers for the three examples. Note that the ranges of the coordinates are still defined within 0 and 100 and each circular dot representing a retailer and a square dot for a supplier. In case 3, there are 25 retailers and a supplier locating at one of the four distinct locations.

Table 3. The primary features of the FLSFSSP examples.

Case	Number of Retailers	Maximal Number of Factories	Number of Suppliers	The optimal value
Case 1	100	4	3	Unknown
Case 2	200	8	6	Unknown
Case 3	100	4	4	53185.21

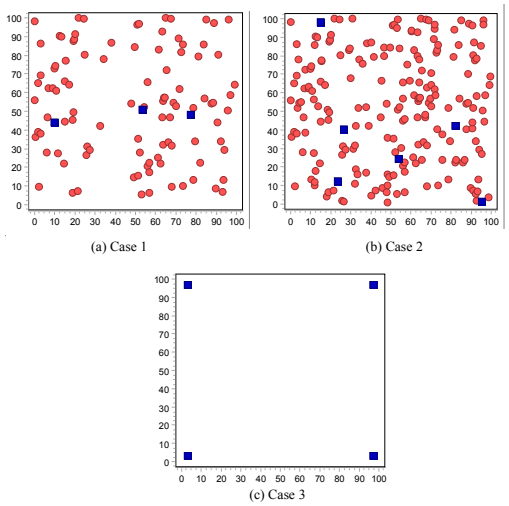

Figure 11. Distribution of retainers and suppliers of the three examples.

The parameter settings for the three methods are the same as those specified in solving the FLSSPs. The objective values obtained from the three methods are shown in Table 4.

Table 4. Objective values of the FLSFSSPs obtained from the three methods.

Case	GA	MFGA	AGGAHM
Case 1	53955.03	53734.35	53171.26
Case 2	107270.24	109507.34	102278.85
Case 3	60810.76	59491.66	59341.06

Notice that the objective values obtained from the AGGAHM are again the best ones among the three methods. Figure 12 illustrates the objective value variations of the three methods. As shown in the figure, the convergence of the objective value of the AGGAHM is quicker than others.

4.2 Economics Scale Effect on the Supply Chain Design

In this section, three FLSSP examples with different economics scales are constructed to analyze the scale effect on the supply chain design. The specifications of these examples are listed in Table 5. Note that the parameters of the power cost function are different; i.e., the production and procurement cost evaluations are different. The interest is to identify the influence of the cost evaluation on the factory selection for the retailers.

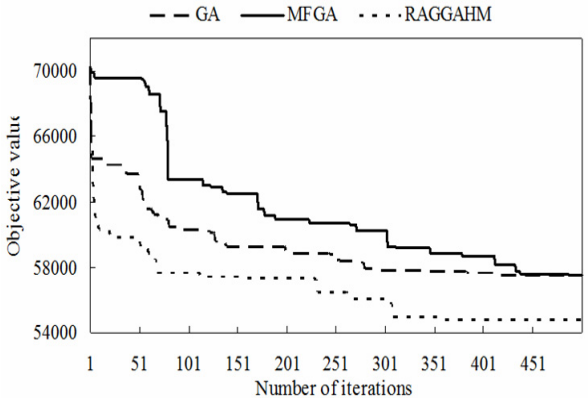

Figure 12. Convergences of the objective values obtained from the three methods.

Table 5. The primary features of the FLSSP examples with different economics scales.

Case	Number of Retailers	Number of Factories	Power cost Function Used
Case 1	100	4	$f(x) = 310x^{0.5}$
Case 2	100	4	$f(x) = 160x^{0.8}$
Case 3	100	4	$f(x) = 110x^{0.95}$

The AGGAHM is used to solve these examples with the same parameter settings described in Section 4.1.1. After 60,000 iterations, the current best factory selections of the examples are shown in Figure 13.

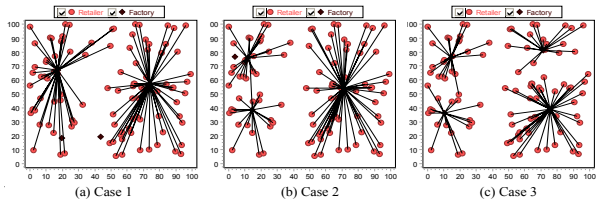

Figure 13. The factory selection comparison between these examples.

In case 1, since the effect of economics of scale is significant, only two factories are needed for the dispersed retainers. Therefore, each factory will produce more products to reduce the production cost. As a result, the retailers form two large groups, with each one centered with a factory. While the effect of economics of scales declines, the production cost reduction is insignificant. To save the transportation cost, more factories are acquired to shorten the transportation distance for the retailers. Consequently, the retailers are grouped into four small groups surrounding a centered factory in case 3. These numerical tests show that the economics of scales plays an important role in supply chain design.

4.3 Effect of Frozen Condition on the AGGAHM

Since the AGGAHM conducts a mutual frozen procedure to alternatively advance the selection gaming results and evolve the coordinates of locations, the setting of frozen condition is of interest. If the number of iterations for the frozen condition is set small, frequent switches of computations make the AGGAHM unable to comprehensively explore the solution space. On the other hand, if the number is too large, the AGGAHM is likely to be tapped within local optima.

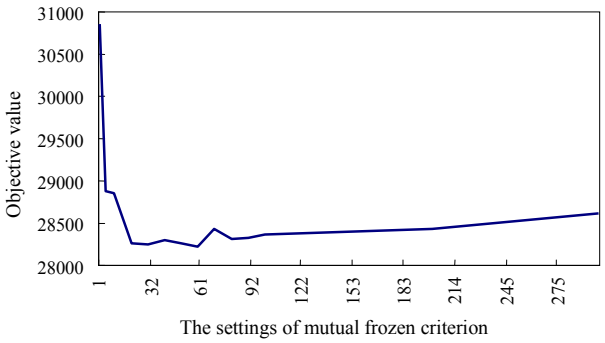

Figure 14. The objective values obtained for frozen conditions with different numbers of iterations.

We test different numbers of iterations of the frozen condition: 1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200 and 300, to observe their objective value achievements. Specifically, a FLSSP with 100 retailers and 4 factories is constructed for the test and the resulting objective values are shown in Figure 14.

As shown in the figure, the solution improves quickly when the number of iterations increased from 1 to 20. Normally, the values between 20 and 60 yield better results. When the number exceeds 60, the solution starts degenerating. Therefore, the best value of the number of iterations without improvement for solving the FLSSP is between 20 and 60.

5. CONCLUDING REMARKS

This paper proposes two factory location setting and factory/supplier selection problems for supply chain designs. To efficiently solve the problems, the agent-based selection gaming computation is introduced. Specifically, an agent gaming and GA operations hybrid method, the AGGAHM, is proposed. The AGGAHM conducts an exclusively and alternatively frozen mechanism to execute either the selection gaming or the coordinate evolving computation. Numerical examples are constructed and tested for three GA-related methods, including the AGGAHM. Numerical results show that the performance of the AGGAHM outperforms other methods. In addition, due to the introduction of agent gaming, the method has a superior convergence speed than others. In addition, numerical tests are conducted to show that the economics of scale plays a significant role in supply chain designs.

REFERENCES

Beamon, B. M. (1998), Supply chain design and analysis: Models and methods, *International Journal of Production Economics*, **55**(3), 281-294.

Chen, J. M., Chen, Y. S., and Chien, M. C. (2008), Optimal lot-sizing and pricing with markdown for a newsvendor problem, *Industrial Engineering and Management Systems*, **7**(3), 257-265.

Chopra, S. and Meindl, P. (2004), *Supply Chain Management: Strategy, Planning and Operations*, Prentica Hall, Upper Saddle River, NJ.

Cohen, M. A. and Moon, S. (1990), Impact of production scale economics, manufacturing complexity, and transportation costs on supply chain facility networks, *Journal of Manufacturing and Operations Management*, **3**, 269-292.

Holland, H. H. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Detroit, MI.

Ishibuchi, H., Sakamoto, R., and Nakashima, T. (2001), Evolution of unplanned coordination in a market

selection game, *IEEE Transactions on Evolutionary Computation*, **5**(5), 524-534.

Luss, H. (1982), Operations research and capacity expansion problems: A survey, *Operations Research*, **30**(5), 907-947.

Mexixell, M. J. and Gargeya, V. B. (2005), Global supply chain design: A literature review and critique, *Transportation Research Part E-logistics and Transportation Review*, **41**(6), 531-550.

Ong, N.-S. and Tan, W.-C. (2002), Sequence placement planning for high-speed PCB assembly machine, *Integrated Manufacturing Systems*, **13**(1), 35-46.

Palisade Co. (2001), *Evolver*, *The Genetic Algorithm Super Solver*, Palisade Corporation, NY.

Prasertwattana, K. and Chiadamrong, N. (2004), Purchasing and Inventory Policy in a Supply Chain under the Periodic Review: A Single Manufacturer and Multiple Retailer's Case, *Industrial Engineering and Management Systems*, **3**(1), 38-51.

Song, S. H. (2006), Multi-Period Integrated Inventory and Distribution Planning with Dynamic Distribution Center Assignment, *Industrial Engineering and Management Systems*, **5**(2), 132-141.

Tesfatsion, L. (2001), Guest editorial: Agent-based Modeling of Evolutionary Economic Systems, *IEEE Transactions on Evolutionary Computation*, **5**(5), 437-441.