# Solving Multi-level Lot Sizing Problem with Memetic Algorithm Based on Refinement Procedure


Haejoong Kim
Seoul National University
ASRI(Automation and System Research Institute)
Dept. of Industrial Engineering
Shillim-dong, Kwanak-ku, Seoul 151-742,Korea
+82-2-880-7182
ieguru@ultra.snu.ac.kr

Sung-Won Jung
Samsung Electronic Company
Memory Division, Semiconductor Business
Hwasung, Gyeonggi, Korea, 445-701,
82-31-208-7701
sungwoni.jung@samsung.com

Jinwoo Park
Seoul National University
ASRI(Automation and System Research Institute)
Dept. of Industrial Engineering
Shillim-dong, Kwanak-ku, Seoul 151-742,Korea
+82-2-880-7182
autofact@snu.ac.kr



## ABSTRACT

Production planning is a core function in manufacturing systems and is gaining even greater attention in supply chain environments where many mutually dependent and cooperative manufacturers are involved. Lot sizing is one of the most important and difficult problems in production planning. While optimal solution algorithms exist for this problem, only very small problems can be solved in a reasonable computation time because the problem is NP-hard.

In this paper we present a meta-heuristic approach, which we call "Memetic Algorithm based on Refinement Procedure", to solve multi-level lot sizing (MLLS) problem. We use a local refinement procedure based on benchmarking to facilitate the solution search. The benchmark-based refinement procedure proposed by this study is also applicable to other problems where solutions are difficult to refine.



## Keywords

Multi-level lot sizing, Benchmark-based Genetic Algorithm, Memetic Algorithm based on Refinement Procedure


## 1. INTRODUCTION

Production planning and scheduling is a core function within the management of process and assembly systems for the manufacture of final products from raw materials and component parts. Their importance has been strengthened by the growing attention directed toward supply chain management and ERP (Enterprise Resource Planning) software.

Lot sizing is a segment of the production planning procedure which decides the quantities of what items have to be produced at what time to meet the known (or estimated) demand without backlogs and stockouts. Lot means the quantity of a product manufactured on a machine continuously without interruption, so establishing a large lot can reduce the setup cost by requiring less frequent changeovers. However, a larger lot increases the holding cost because more items which are produced in a period to meet some future demand must be stored in inventory. Thus lot sizing is a trade-off between low setup costs, which favor large production lots, and low holding costs, which favor a lot-for-lot like production.

Table 1 presents an example of a lot sizing problem. The first production lot of size 30 which covers demands up to period 2 is produced in period 1 (a). We also see that items which are produced in advance occupy inventory (b).

**Table 1. An example of lot sizing**

| Phase | t=1 | t=2 | t=3 | t=4 | t=5 |
|---|---|---|---|---|---|
| Demand | 10 | 20 | 20 | 15 | 10 |
| Setup | 1 | | 1 | | |
| Production Lot | 30 | a | 45 | | b |
| Inventory | 20 | | 25 | 10 | |

It is not difficult to find the optimal production lot when only an end item is considered (single-level case). However in a real production system in which complex product structures are taken into account, a production plan must respect the precedence relationships of operation. When multi-level structures are taken into account, this problem is termed multi-level lot sizing (MLLS). MLLS problems have greater importance under the environment in which production plants in a supply chain can cooperate to make a production plan for the master production scheduling (MPS) of each facility unit in the aspect of supply chain optimization. Figure 1 shows the environment of an MLLS problem.

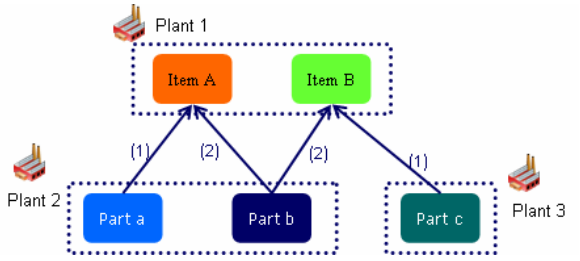


**Figure 1. Environment of an MLLS problem**

Material Requirements Planning (MRP) is the most popular lot sizing procedure used in industry for planning the production of the end items, as well as the production (or purchase) of its components. By taking the anticipated build schedule from the MPS and ‘‘exploding’’ it through the Bill of Materials, the purchase and/or production of the items and its components is established, in order to create a demand forecast [2]. However, MRP is not enough, as its basic philosophy is only to ensure that the right number of components is planned at the right time to meet the demand for the end items. MRP therefore only provides a feasible solution to the MLLS problem.

Optimal solution algorithms exist for this problem, but only very small problems can be solved in reasonable computation time for the problem is NP-hard, not to mention the mathematical complexity of the technique that might deter many potential users. Several approaches have been developed to solve variants of the MLLS problem, with further assumptions made on the product and/or cost structure, but execution times remain excessively high for application to real problems. Hence heuristic techniques that offer a reasonable trade-off between optimality and computational feasibility are highly desirable [3].

The main objective of this study is to reduce the execution time while maintaining a relatively good quality production plan, so we assume the simple MLLS problem. The demands for the final products in the supply chain are assumed to be deterministically known by forecasting. The demand for components is derived from the production plan for final products. It is also assumed that no backlogging is allowed for all component parts and final products and that no external demand for components is allowed. For the sake of simplicity, we assume that neither positive initial inventories nor scheduled receipts are introduced. Furthermore, it is assumed that linear holding, production and transportation costs occur for carrying inventory and producing items, respectively. Setup cost is incurred if there is a positive produced quantity in a given period. We also assume that there are no more than two production facilities which produce the same item.

This text is organized as follows. Section 2 presents a literature review about solution approaches to the MLLS problem. A generic MLLS problem is formulated by means of a mixed-integer program in Section 3. Section 4 introduces a memetic algorithm (MA) and Section 5 refines the refinement procedure in our algorithm. In Section 6 a computational study is performed. Section 7 summarizes the work.

## 2. LITERATURE REVIEW

The MLLS problem has been continuously studied since the 1960s. Of the three basic approaches to MLLS that have been described in the literature, one focuses on developing an algorithm to obtain the optimal solution.

Zangwill modeled the MLLS problem for a serial system as a single-source network problem, and devised a dynamic programming recursion to obtain the optimal solution [4]. Love also considered a serial system and proposed an alternate algorithm, which shows the nested property of the solution [5]. Steinberg and Napier expressed the MLLS problem in general systems as a network model and solved the problem using the branch and bound method [6]. However, these methods require significant computational effort for solving large-scale problems.

The second approach using heuristic lot-sizing has started to gain attention in this area, since all these optimization algorithms for MLLS are only applicable to unrealistically small scale problems.

Yelle et al. suggested a sequential approach method which ignores the interdependencies between stages of the process and applies a single-stage lot-sizing method sequentially. This suboptimal approach, while computationally simple, may yield substantial cost penalties as it does not consider the interdependency between the items in stages [7]. Blackburn and Millen considered the relationship between each level of the assembly system and suggested some improved heuristics by modifying cost meters to reflect those relationships between each stage [8]. Heinrich and Schneeweiss devised a stage by stage heuristics based on the concept of the modification of the cost parameters suggested by Graves [9].

The third approach is to use the meta-heuristic method which has shown very attractive performance in many combinatorial problems. Kuik and Salomon applied the simulated annealing search method to the MLLS problem [10]. Their experiments showed that good solutions can be found by simulated annealing in a reasonable amount of time. They also suggested incorporating more problem-specific features in designing an efficient, simulated annealing, search method for future research. Recently, Dellaert and Jeunet have proposed a heuristic method based on the genetic algorithm (GA) which can be applied to the MLLS problem for general product structures [3]. They developed a binary encoding GA and designed some genetic operators. Although they demonstrated the efficiency of their solution method in their research, further refinements are still possible in light of further improvement.

The primary purpose of this research is to propose new methods to improve the previous cost-effective evolutionary algorithm in order to provide higher quality solutions to large-scale problems which occur in real manufacturing environments.

## 3. MATHEMATICAL FORMULATION

The following, well-known, mixed-integer model gives a precise specification for the MLLS problem.

$$\text{Min} \sum_{i=1}^{P} \sum_{t=1}^{T} (s_{i,t} y_{i,t} + h_{i,t} I_{i,t} + p_{i,t} x_{i,t}) \quad (1)$$

Subject to

$$I_{i,t} = I_{i,t-1} + x_{i,t} - d_{i,t} \quad (2)$$

$$d_{i,t} = \sum_{j \in \Gamma(i)} c_{i,j} x_{j,t+l_{i,j}} \quad , \quad \Gamma(i) \neq \phi \quad (3)$$

$$x_{i,t} - M y_{i,t} \leq 0 \quad , \quad y_{i,t} \in \{0,1\} \quad (4)$$

$$I_{i,t} \geq 0 \ , \ x_{i,t} \geq 0 \quad (5)$$

**Cost parameters**

| | |
|---|---|
| $s_{i,t}$ | Set-up cost for item i in period t |
| $h_{i,t}$ | Unit inventory carrying cost per period for item i in period t |
| $p_{i,t}$ | Unit production or purchase cost for item i in period t |

**Decision variables**

| | |
|---|---|
| $d_{i,t}$ | Gross requirements for item i in period t |
| $x_{i,t}$ | Production or purchase quantity of item i at the beginning of period t |
| $I_{i,t}$ | Level of inventory for item i at the end of period t |
| $y_{i,t}$ | Boolean variable addressed to capture i's set-up cost in period t |

**Technical symbol & coefficients**

| | |
|---|---|
| $l_{i,j}$ | Lead time required to deliver item j to item i |
| $c_{i,t}$ | Quantity of item i required to produce one unit of item j |
| $J$ | Number of items. |
| $T$ | Number of periods. |
| $\Gamma(i)$ | A set of immediate successors for item i |

The objective function (1) minimizes the total sum of purchase or production costs, setup and inventory holding costs for all items over the planning horizon. The constraint (2) is the conservation equation for items in the inventory, and the constraint (3) calculates the required component item i for production of its direct predecessor item j. Notice that the demand for end items in each time period is assumed to be predetermined as mentioned above. Constraint (4) assures that a setup cost will be incurred when a lot size is produced. Finally, constraint (5) states that backlog is not allowed and that production is either positive or zero.

The main objective of this paper is to present heuristic methods based on evolutionary algorithms to address the MLLS problem, including setup costs and setup times. More specifically, we developed a MA, an evolutionary algorithm that makes use of local search techniques. This technique has been shown to be highly effective for several problems in production planning and scheduling

# 4. THE PROPOSED MEMETIC ALGORITHM(MA)

## 4.1 Representation Scheme

An MA is an evolutionary algorithm, similar to the GA, which is based on a population of agents. However, the method is less constrained than GA, since it does not use any biological metaphor that would restrict the design of its components. As a consequence, each agent can make use of previous knowledge of solution results [11].

To design a GA for a particular problem, we first need to devise a suitable representation scheme that shows the solution characteristics. In the application of GA to lot sizing problems, previous researchers usually defined the setup status called 'meme' [3] [10]. Let us assume that production quantities and inventories for the item i at time period t are given respectively. The property known as "zero-switch" indicates that an optimal solution exists for MLLS, which enables the effort to contrive the encoding scheme for MLLS to be reduced. The setup status information alone for all items over the planning period in each facility is enough to generate a perfect production plan. Let us assume that the meme yi,t represents the setup status in which it takes value 1 if a setup for item i in time period t is introduced and 0 otherwise. Figure 2 shows the meme representation.

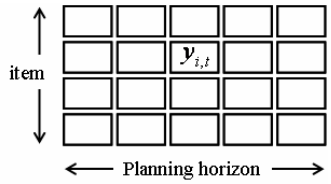


**Figure 2. Meme representation**

## 4.2 Memetic agent

Each memetic agent maintains the information structure shown in figure 3 which represents the solution for MLLS in supply chain management. Meme represents the core information to establish the solution. Through the decoding and repairing procedure, meme information is transformed into a specific production plan comprising production quantities, inventories and demands for all items over the predicted planning horizon in each facility. Based on the decoded production plan, the total cost, consisting of inventory, setup, production and transportation costs incurred for all the items over the planning horizon, can be calculated.

```
Struct memetic_agents {
    %meme information
    meme[max_item] [max_time]

    %production information
    production_qty [max_item] [max_time]
    demad_qty[max_item] [max_time]
    inventory_qty[max_item] [max_time]

    %cost information
    setup_cost[max_item] [max_time]
    inventory_cost[max_item] [max_time]
    production_cost[max_item] [max_time]
    Total_Cost

    %Fitness Value information
    Fitness_Value (f)
}
```

**Figure 3. Memetic agent**

## 4.3 The procedure of the memetic algorithm

Figure 4 shows the procedure of our proposed MA. In the first step, a number of memetic agents are generated, each of which has meme information which represents a specific solution for a given problem. This meme information is improved to generate the high quality solution through a recombination, mutation and

local refinement procedure. Whenever the meme information of each memetic agents is considered convergent, some of the memetic agents are replaced by new agents whose meme information is generated randomly. These procedures are iterated until they meet the termination criterion. In the next subsection, we explain each procedure in more detail.

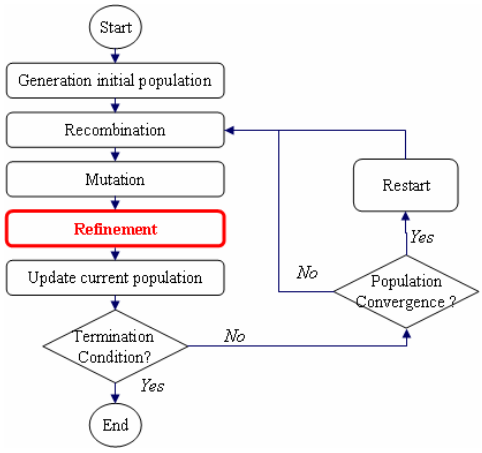


**Figure 4. The procedure of the memetic algorithm**

### *4.3.1 Recombination procedure*

The recombination procedure plays a similar role to that of the crossover operation in GA. In the recombination procedure, we select two agents using a roulette wheel method, determine a specific time period as a criterion and exchange the meme information based on the determined time period. Two child agents are introduced, after which the best among those four agents is selected and put it in the new agent set. This process is iterated until the population size of the new agent set reaches that of the current one.

### *4.3.2 Mutation procedure*

Mutation has the general purpose of introducing new information not possessed by the parent solution set which thereby expands the search space. A random set of 0 and 1 is created for selected meme based on the predetermined mutation rate. This study selects the non-uniform mutation. It gradually increases the mutation rate to the predetermined point as the algorithm proceeds and thereby delays the convergence speed of the solution with the unfolding evolution.

### *4.3.3 Refinement procedure*

As mentioned above, we devise the benchmark procedure as the refinement for the MLLS problem. This is the most important procedure in our proposed MA which is described in detail in the following section.

## 4.4 Refinement procedures based on benchmarking procedure

### *4.4.1 Basic concept*

We have noted above that lot sizing is a trade-off between setup costs and holding costs. Increasing the lot size increases the average amount of inventory on one hand, but on the other reduces the frequency of setup. Therefore, we use this concept as a refinement procedure using benchmark.

For simplicity, let us consider the solution search process which can be found in a probabilistic search method for the single level lot sizing problem in Figure 5. Let current setup frequency be $S^a$. We can reduce holding cost by increasing setup frequency, which contrarily increases setup cost, and which ultimately reduces the total cost. It is possible that total cost is better at setup frequency point $S^a$ than at point $S^*$ where the optimal solution exists.

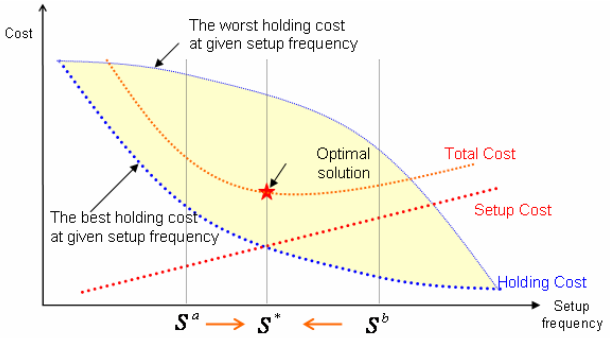


**Figure 5. Refinement concept in single-level lot sizing**

In a probabilistic search process, the randomly-created solution approaches the value of the near-optimal number of setups, thereby deriving a superior solution by finding the best decision for the setup point with the near-optimal number of setups. If we can find a method capable of reaching the near optimal number of setups in a level lot sizing problem, we can narrow down the search area for the solution by inducing searches to the area for the setup point given the number of setups. In production plans with a single level, the near optimal number of setups can be derived by using the EOQ (Economy Order Quantity) or Silver-Mill method. The optimum solution can be found in a short time, even if it is a large scale problem, by inducing the search area around the number of setups obtained by these methods.

We next consider the integrated production planning in a multi-level product group such as in Figure 6. According to the plan, the total cost will be the sum of all costs derived from the production plans of each item. If these products have no relationship with each other in production planning, the total cost for the integrated production planning can be minimized by minimizing the total cost of the production plan for each product.

However, it is hard to find the optimal number of setups for each product, since the change of number of setups in the upper-level products will affect the demand pattern for lower-level products which is directly connected to them.

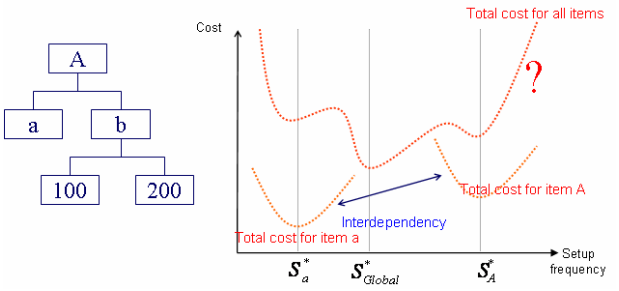


**Figure 6. The difficulty of finding the optimal setup frequency in multi-level lot sizing**

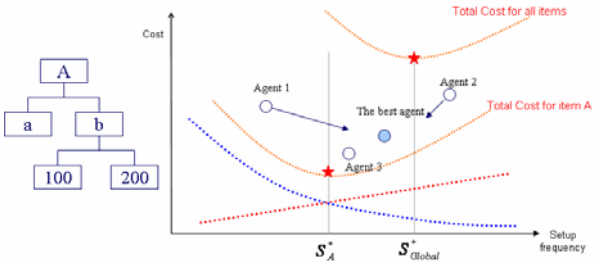


**Figure 7. Refinement procedure at the level of each item**

Let us define the optimal setup frequency for item A as $S^*_A$ if the plan for product A is only considered, and $S^*_{global}$ if the optimization of overall integrated production planning is considered. If we apply EOQ or Silver-Mill methods for the refinement processes, it will focus the solution search area around the local optimum number of setups, $S^*_A$. To find the global optimum number of setups, we need to restrict the search area to around the global optimum number of setups, $S^*_{global}$. However, due to the complexities in product relationship, no effective refinement method has yet been elucidated.

Based on above concept, we propose two refinement procedures based on benchmarking which can be applied with MA. The first refinement-method uses the number of setups directly as the criterion for benchmarking to improve the solution. The second method uses the ratio of the holding cost to the setup cost which improves the solution better than the first one.

### *4.4.2 Refinement procedure using the number of setups*

In this refinement method, the search area to find the global optimum number of setups is restricted to around the current best number of setups. The number of setups can be controlled at the level of each item or total number of setups over all the items.

When the number of setups is controlled at the level of each item, the holding and setup costs are compared with those of the best solution, then the movement is determined independently for each item. Some items will increase the setup occurrence while others will decrease it. We next consider refining the production plan of an upper level product A in agents 1, 2 and 3, as shown in figure 7. When comparing the agents of the production plan, if we consider the effects on product A only, the setup cost increases in agent 1 while the inventory holding cost increases in agent 2. The production plan for product A generated by these agents incurs more cost than that of the best agent. The number of setups in agent 3 is closer to $S^*_A$ than the best agent and the production plan for item A is therefore cheaper than that of the best agent.

When applying the refinement process based on benchmarking, the number of setups for item A of agent 1 should be increased since the holding cost for item A of agent 1 is higher than that of the best agent. Agent 2 is the opposite – it creates more setup holding costs, so the number of setups should be decreased. Agent 3 incurs less cost than the best agent if we consider the production plan for item A only, so that no modification is performed at that time. However, the solution quality for item A of agent 3 will be degraded through evolutionary processes such as recombination and mutation, and it will therefore benchmark the number of setups of the best agent later.

When the total number of setups is applied to the refinement procedure, only total setup is controlled. We next consider refining the production plan in agents 1, 2 and 3 in figure 8. While we keep the best agent, agents 1, 2 and 3 will move (reduce or increase) the number of setups toward the best agent. The number of setups in agents 1 and 2 should be increased since total cost of agent 1 is higher and the number of setups is lower than the best agent. The setup of agent 2 will be modified because the total cost is higher and the number of setups is lower than the best agent.

In this method, it's very important to decide which item will add or delete a setup and how many setups are modified. We consider the relative difference in the number of setups. Figure 8 shows this process. The difference in the number of setups between agent 1 and the best setup is more than between agent 2 and the best agent. Therefore, agent 1 will add more setups than agent 2. The item which needs modification for the number of setups is selected by the relative setup frequency factor called Ri. The score is increased if the difference in the number of setups compared with the best agent increases.

*Ri(Relative setup frequency i)= the number of item i in agent n – the number of item i in the best agent.*

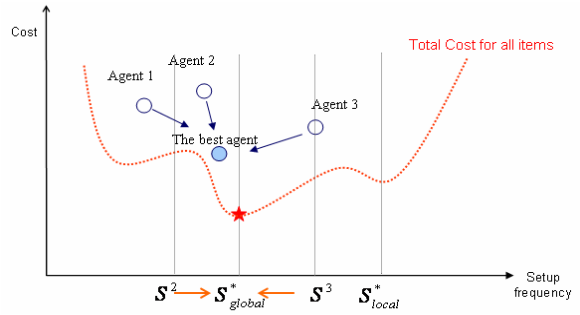


**Figure 8. Refinement concept in multi-level lot sizing**

Figure 9 shows the refinement procedure using the number of setups at the level of each item.

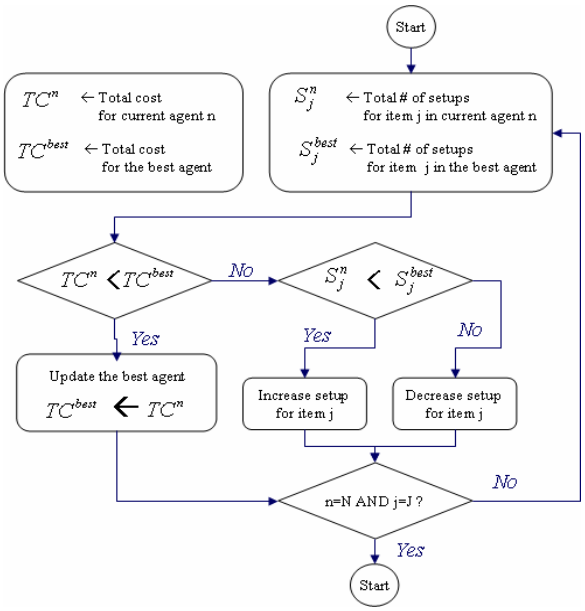


**Figure 9. Refinement procedure using the number of setups**

### *4.4.3 Refinement procedure using the ratio of setup cost to holding cost*

This procedure gives an improved result from refinement methods merely by using the number of setups. As repetitive experiments are made we notice that refinement methods using the number of setups tend to become easily trapped into the local optimum easily, with the consequence that when the number of setups is applied to the refinement procedure directly, the relationship between the number of setups and total cost may cause many local optimums.

Figure 10 shows one of the experimental results when the refinement procedure using the number of setups as benchmarking is applied. It is apparent that the total cost according to the number of setups has presents the intended shape, indicating the presence of many local optimums.

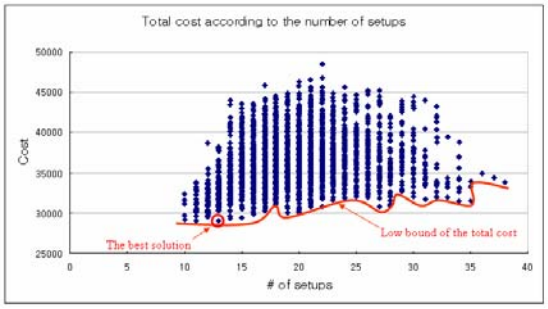


**Figure 10. An experimental result of the refinement procedure using the number of setups**

Instead of applying the number of setups or setup cost directly, we use the ratio of holding cost to setup cost as the criterion for benchmark. Data in figure 10 are transformed into the relationship between the total cost and the ratio of setup cost to holding cost in figure 11.

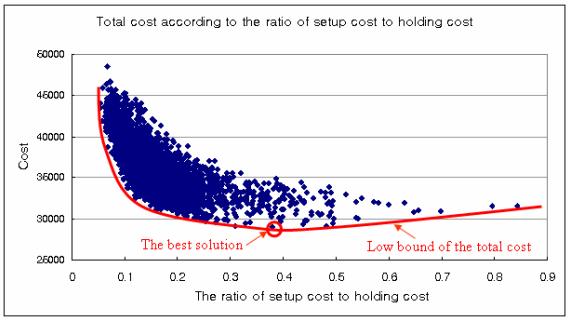


**Figure 11. An experimental result of the refinement procedure using the ratio of setup cost to holding cost**

It seems that the lower bound of total cost draws a single large concave, or if not, it has few local optima. This demonstrates that this method will reduce the probability to be trapped in the local optimum. Figure 12 shows the refinement procedure using the ratio of setup cost to holding cost.

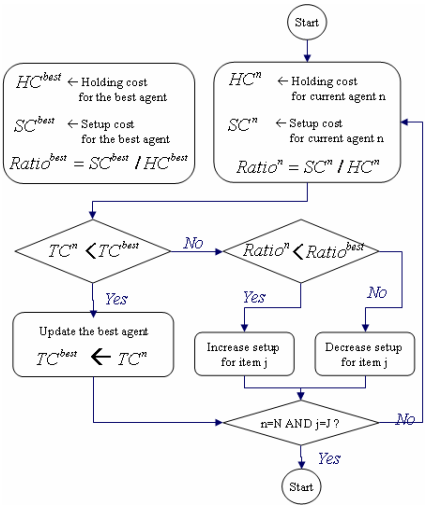


**Figure 12. Refinement procedure using the ratio of setup cost to holding cost**

# 5. COMPUTATIONAL EXPERIMENT

## 5.1 Experimental design

Due to the difficulty of obtaining optimal solutions for MLLS problems involving general product structures, we limited the study to a performance comparison of the selected heuristics for assembly and general product structures of large size.

The individual problems were generated by combining i) the size of the problem, ii) the commonality index (C value) and iii) the time between order (TBO) value.

The size of the problem was determined by the depth of supply chain, the number of items and the length of the planning horizon. The C value was proposed by Collier and is defined as the average number of successors per component. [12]:

$$C = \sum_{i=F=1}^{P} \Gamma(i) / (P - F) \qquad (12)$$

Let h and D be the inventory holding cost and the average demand for the item being considered, respectively. TBO can be used to deduce the set-up cost in the following way.

$$TBO = \sqrt{\frac{2s}{hD}} \rightarrow s = \frac{TBO^2 \cdot h \cdot D}{2} \qquad (13)$$

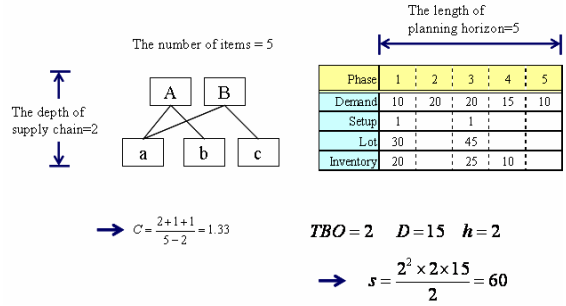


**Figure 13. An example experimental computation**

We tested our MA considering the instance of two-stage supply chain problems which comprise 5 final items and 10 components with varying planning horizons. We set TBO as 2 periods for all items and assigned the structure of the product to satisfy C as 2. The inventory cost for each item was computed to reflect the value-added holding cost. The demand for end items was determined from a uniform distribution over [0, 20].

We created 20 test problems by only varying the demand per each period. We obtained the solutions by GA and MA with the refinement procedure using the number of setups and the ratio of setup cost to holding cost.

**GA**: Genetic algorithm

**MA-N**: memetic algorithm with refinement procedure using the number of setups

**MA-R**: memetic algorithm with refinement procedure using the ratio of setup cost to holding cost

Although a more precise experimental design is required for a better comparison between the algorithms, we were limited to this simple test due to the lack of computation time. This paper has focused on comparing refinement procedures, which will be proven by the following experimental result. In a future study we will test the additional instances with a more precise design.

## 5.2 Results

Figure 14 is a good example showing the difference of performance evolution according to the generation by each solution method. The reduction of costs obtained by MA-N and MA-R is faster and better than that by GA. This means that the probabilistic search methods can improve the search speed using the criteria which informs the route of search. It is apparent that MA-R is better that MA-N, i.e., the refinement procedure using the ratio of setup cost to holding cost is better than that using the number of setups.

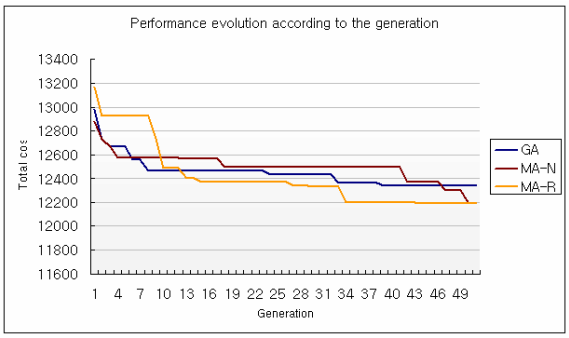


**Figure 14. Performance evolution according to the generation**

Table 2 shows the statistical results of 20 test problems. The results from MA-R and MA-N are better than from GA. MA-R is superior to MA-N. However, there remains the possibility that GA or MA-N is better than MA-R.

**Table 2. The statistical results of 20 test problems**

| | GA | MA-N | MA-R |
|---|---|---|---|
| Average | 12450 | 12312 | 12258 |
| std | 277 | 312 | 288 |

## 6. CONCLUSION

In this study, we developed a heuristic for integrated master production planning in supply chain optimization based on MAs and devised a local refinement procedure based on the concept of benchmarking, which guarantees the evolution of solutions by taking into account the interrelationships among memes. The refinement procedure proposed in this study uses the number of setups and the ratio of setup cost to holding cost as its basis for benchmarking. Each memetic agent benchmarks its criterion of the best agent in the population to modify the meme information and thereby improve the solution quality.

The effectiveness of the proposed algorithm was tested through a series of simulation experiments with various problem sizes. Comparing the solutions generated from several simulation experiments by the proposed algorithm with those from GA, those from MAs with the refinement procedure were better than those from the ordinary GA. For a more accurate comparison between the proposed methods, future study will require more precise experimental design and a greater number of test cases.

We believe that this proposed algorithm based on benchmarking can be applied to other, difficult-to-solve problems. The refinement procedure will form the basis for improving MLLS solutions in supply chain management and ERP system. There is a need for further study in order to improve the speed of MAs by enhancing the refinement procedure and to generate a more efficient meme design.

## 7. ACKNOLEDGEMENT

This research was supported by the project, 'Development of a Knowledge-based Collaborative Manufacturing System', one of the 'Next Generation New Technology Development' programs funded by the Ministry of Commerce, Industry and Energy (MOCIE), Republic of Korea. We would also like to acknowledge the Automation and Systems Research Institute (ASRI) in Seoul National University for providing resources and administrative support.

## 8. REFERENCES

[1] A. Drexl, A.Kimms. Lot sizing and scheduling – Survey and extensions. *European Journal of operational research. 99* (1999), 221-235.

[2] Regina Berretta, Luiz Fernando Rodrigues. A memetic algorithm for a multistage capacitated lot-sizing problem. *Int. J. Production Economics* 87 (2004) 67–81.

[3] N. Dellaert, J. Jeunet, N. Jonard. A genetic algorithm to solve the general multi-level lot-sizing problem with time-varying costs. *Int. J. Production Economics* 68 (2000) 241–257.

[4] Zangwill W.I. A backlogging model and a multi-echelon model of a dynamic economic lot size production system-a network approach, *Management Science* 15 (1969) 506-527

[5] Love, S.F. Facilities in series inventory model with nested schedules. *Management Science* 18 (1972) 327-338

[6] Steinberg, E. and Napier A., Optimal multi-level lot sizing for requirements planning systems, *Management Science* 26 (1980) 1258-1271

[7] Yelle, L.E. Materials requirements lot sizing: A multi-level approach. *International Journal of Production Research* 17 (1979) 506-527

[8] Blackburn, J.D. and Millen, R.A., 1982, Improved heuristics for multistage requirements planning systems, *Management Science* 28 (1982) 44-56

[9] Heinrich, C.E. and Schneeweiss, C. Multi-stage lot-sizing for general production systems, *Springer Berlin* (1986)

[10] Kuik, R., Salomon S. Multi-level lot-sizing problem: evaluation of a simulated-annealing heuristic. *European Journal of Operational Research* 45 (1990) 25-37

[11] Regina Berrettaa, Luiz Fernando Rodrigues. A memetic algorithm for a multistage capacitated lot-sizing problem. *Int. J. Production Economics* 87 (2004) 67–81

[12] Collier, D.A.. The measurement and operating benefits of component part commonality. *Decision Sciences* 12 (1981) 85–96.