Contemporary Management Research
Pages 313-328, Vol. 7, No. 4, December 2011

# Using an Efficient New Gene for Genetic Algorithm to Solve the Multi-buyer Joint Replenishment Problem

Chun-Wei R. Lin
National Yunlin University of Science and Technology
E-Mail: lincwr@yuntech.edu.tw

Hsian-Jong Hsiau
National Yunlin University of Science and Technology
E-Mail: g9421805@yuntech.edu.tw

## ABSTRACT

The multi-buyer joint replenishment problem (MJRP) is the multi-item inventory problem which deals with the replenishment of a group of product items that are jointly delivered to multi-buyer. The objective of MJRP is to develop policy to minimize the total cost which consists of the holding cost and the transport cost. In this paper, we propose a modified genetic algorithm (called GAT) which adopts a New Gene, basic cycle time, to solve the multi-buyer joint replenishment problem (MJRP). The genetic algorithm (GA) has been widely applied to solve MJRP. However, most of the literature which used the genes for chromosomes were the ratio of each product delivery cycle time to the basic cycle time. This searching method is called GAK here. The disadvantage of GAK is that the number of genes is determined by the number of product items and buyers, and the length of chromosomes will be expanded when the number of product items or buyers is increased. The length of chromosomes will impact the CPU running time in the genetic algorithm. The proposed GAT can improve the disadvantage of GAK. Simulation experiments demonstrate that the GAT is very efficient and outperforms GAK. The running time of GAK is improved over 96% by GAT.

Keywords: Joint Replenishment Problem, Multi-item Multi-buyer, Genetic Algorithm

## INTRODUCTION

In the real world, usually products are jointed to ship to multiple buyers or retailers by suppliers in order to reduce costs such as on gasoline and petrochemical products distribution. The supplier has to consider the holding cost of a product and the transport cost to minimize the total cost. In the classical EOQ model without considering joint replenishment, optimal ordering size and basic cycle time can be computed based on the demand of buyers. But when delivery items are considered to be jointed, the problem is termed as the joint replenishment problem (JRP). The JRP is a NP-hard problem proven by Arkin et al. (1989).

The JRP has been extensively studied during the past two decades. Many iterative solution techniques of JPR have been presented in the literature. Kaspi and Rosenblatt (1991) proposed a RAND approach for solving the economic ordering quantity for jointly replenished items. The authors conducted an extensive simulation study and concluded that the RAND solution procedure outperforms previously known algorithm (Goyal 1974) for solving this problem. They claimed that using RAND 10 compared to optimal (full enumeration) solution, the maximum error value found is less than 0.2%. Goyal et al. (1993) suggested a modified procedure which used a better estimate for the lower bound of the basic cycle time. Wildeman (1997) presented an efficient optimal solution method for JRP by applying Lipschitz optimization to obtain a solution with an arbitrarily small deviation from the optimal solution. Frenk et al. (1999) proposed an efficient algorithm with 'Improved-feasiblity procedure' to determine the optimal policy of the multi-item and one-buyer inventory problem. Khouja et al. (2000) used 1200 randomly generated problems to make a comparison between genetic algorithms and the RAND method for solving the JRP. The GA found the solutions with the same total cost as the RAND for 761 problems, outperformed the RAND for six problems and under-performed the RAND for 433 problems. The maximum percentage savings in total cost provided by the RAND was 0.078%. Olsen (2005) developed an evolutionary algorithm (EA) that used a direct grouping method to solve JRP. The authors showed that EA for joint replenishment policy incurs a lower total cost than the best available algorithm for some parameters of factor. Overall the EA improved over the RAND for 5.1% of the 72,900 randomly generated problems. The maximum percentage savings in total cost provided by the RAND was 0.69% from the 72,900 problems.

Recently, Moon et al. (2006) developed two algorithms for solving JPR with resource restriction. The main resource restriction in their paper is capital that can be invested. One of the algorithms is to modify the existing RAND algorithm to be

applicable to this problem and another is to develop a genetic algorithm for the JPR with resource restriction. Hoque (2006) developed a generalized global optimal solution algorithm of the JPR extended model which includes some practical issues

Most of the literature has considered the JRP as a problem of coordinating the replenishment of multi-items for a single buyer. But a common practice for a supplier is to have multi-buyers who order multi-items from the supplier. In this case, the JRP becomes a multi-item multi-buyer joint replenishment problem. MJRP has received little attention in the literature until recently. Chan et al. (2003) proposed a new modified genetic algorithm for solving the MJRP. The performance test was compared to the algorithm of Goyal (1974) and got a better result. Li (2004) proposed a modified RAND method to solve the MJRP, but without a performance test. Chan et al. (2006) addressed the delivery scheduling issue for MJRP, once the optimal replenishment cycles are determined, by formulating four problems according to four different objectives in cost and resource minimization. The authors used the solution of the optimal replenishment cycles which had been solved to do the post research for optimal scheduling.

MJRP is a MINLP problem. In MJRP, the delivery cycle time denoted by $t_{ij}$ is the time interval between two consecutive delivered product items $i$ for buyer $j$. The delivery basic cycle time denoted by $T$ is the minimum value of $t_{ij}$ and $t_{ij}$ is an integer multiple of $T$, i.e. $t_{ij} = K_{ij}T, i = 1,2,..., I, j = 1,2,..., J, \forall K_{ij} \in \text{Integer}$. The decision variables of optimizing total costs include the $K_{ij}$ and $T$, therefore the number of decision variables is $I \times J + 1$, and the problem size depends on the items $I$ and buyers $J$. It is not easy to directly obtain the optimal solution for a large size problem. In previous reviews, the scholars have applied the genetic algorithm (GA) to solve JRP or MJRP. However, they used $K_{ij}$ as the genes to form chromosomes. This method is called GAK here. Since chromosomes are formed by $K_{ij}$, and $K_{ij}$ are the decision variables, the number of $K_{ij}$ is $I \times J$. The length of chromosomes in GAK will be expanded when the number of product items or buyers is increased. The computing time of GAK will be longer when the size of the problem becomes large. The computing time is an important criterion to judge the performance of searching techniques.

In this paper, using a New Gene $T$ is proposed to replace the genes $K_{ij}$ for the genetic algorithm. The proposed method is called GAT. The New Gene $T$ in MJRP is the delivery basic cycle time. In GAT the number of genes $T$ is only one, but in GAK the number of genes $K_{ij}$ is $I \times J$. The chromosomes of GAT with gene $T$ are unlike the chromosome of GAK with the genes $K_{ij}$, the length of chromosomes will

NOT be expanded when the problem size enlarges.

## PROBLEM AND MODEL FORMULATION

### Problem Statement

A single supplier supplies multi-items of $I$ products to multiple buyers, and the number of buyers is $J$. A supply network of jointed replenishment distribution is shown in Figure 1. All consignment inventories belong to the supplier. Each unit of product item $i$ for buyer $j$ in consignment costs the supplier $C_{ij}$ and the holding cost is $h_{ij}$ percent of $C_{ij}$. A fixed common carrier $S$ is charged and a delivery cost $s_{ij}$ of product item $i$ for buyer $j$ would continue to be added per delivery. Each buyer $j$ has a different demand of item $i$ product $D_{ij}$. The goal of this problem is to minimize total costs which include the holding cost and transport costs. The transport costs consist of two components, one is common carrier cost and another is delivery cost.

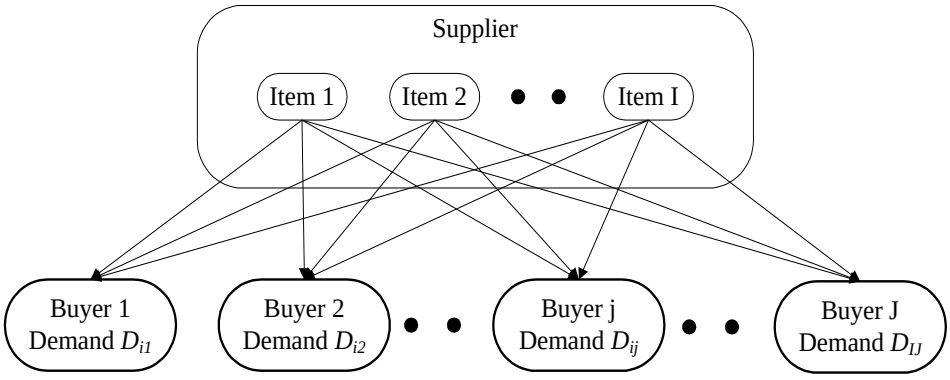


Figure 1 A supply network of jointed replenishment distribution

### Assumptions

The basic assumptions in solving this problem are made as follows:

1. The demand for the each buyer is constant.
2. The holding cost is a fraction of the product cost.
3. The common carrier cost is constant
4. Any product in the set of products can be jointed to transport.
5. Stock-outs are not allowed.

**Notations**

| | |
|---|---|
| $P$ | The set of products, $P=\{1,2,....,I\}$, |
| $B$ | The set of buyers, $B=\{1,2,....,J\}$, |
| $I$ | The number of product items, |
| $J$ | The number of buyers, |
| $D_{ij}$ | Demand of item $i$ for buyer $j$, |
| $h_{ij}$ | Inventory holding cost of item $i$ for buyer $j$, per unit per unit time, |
| $C_{ij}$ | Product unit cost of item $i$ for buyer $j$, |
| $S$ | Common carrier cost, |
| $s_{ij}$ | Delivery cost of item $i$ for buyer $j$, |
| $T$ | Delivery basic cycle time (Decision variable), |
| $t_{ij}$ | Delivery cycle time of item $i$ for buyer $j$, |
| $K_{ij}$ | The ratio of $t_{ij}$ to $T$, $K_{ij}=t_{ij}/T$, $K_{ij}$ is an integer (Decision variable), |
| Z | Total costs consist of holding cost, common carrier cost and delivery cost. |

**Model Formulation**

The goal of the joint replenishment policy is to minimize the total costs which include the holding cost and transport costs. The transport costs include two components, one is the common carrier cost and another is the delivery cost. Suppose the delivery cycle time of item $i$ for buyer $j$ is $t_{ij}$, the holding cost of item $i$ for buyer $j$ is equal to $h_{ij}C_{ij}D_{ij}t_{ij}/2$. Suppose each common carrier cost is $S$ and the delivery basic cycle time is $T$, then the annual common carriers cost is $S/T$. Suppose the delivery cost of item $i$ for buyer $j$ is $s_{ij}$, then the annual delivery cost of buyer $j$ is $s_{ij}/t_{ij}$. Total costs =annual holding cost + annual common carriers cost + annual delivery cost. The objective function can be written as

$$Min\ Z=\sum_{i=1}^{I}\sum_{j=1}^{J} h_{ij}C_{ij}D_{ij}t_{ij}/2+S/T+\sum_{i=1}^{I}\sum_{j=1}^{J} s_{ij}/t_{ij} \tag{1}$$

To reduce total costs, product jointed delivery policy is considered. Thus $t_{ij}$ should be the integer multiple to $T$, i.e. $t_{ij}=K_{ij}T, K_{ij}\in Integer$. The objective function can be rewritten and the constraints are shown as

$$Min\ Z = \frac{T}{2}\left(\sum_{i=1}^{I}\sum_{j=1}^{J} h_{ij}C_{ij}D_{ij}K_{ij}\right) + \frac{1}{T}\left(S + \sum_{i=1}^{I}\sum_{j=1}^{J} s_{ij}/K_{ij}\right) \quad (2)$$

$$s.t.$$

$$K_{ij} \in \text{int}\ ,\ \forall i \in P,\ j \in B \quad (3)$$

In above MJRP model, all the $K_{ij}$ must be positive integers. It is a nonlinear integer problem.

**The Bound of Basic Cycle Time**

The optimal basic cycle time can be derived from (2) by $\partial Z/\partial T = 0$

$$T^* = \sqrt{\frac{2(S + \sum\sum s_{ij}/K_{ij})}{\sum\sum h_{ij}C_{ij}D_{ij}K_{ij}}} \quad (4)$$

$T^*$ is a function of $K_{ij}$ and $K_{ij}$ are the decision variables of $T^*$. The lower bound of $K_{ij} = 1$ . The upper bound of $T^*$ can be decided when $K_{ij} = 1,\ \forall i \in P,\ j \in B.$

$$T_{\max} = \sqrt{\frac{2(S + \sum\sum s_{ij})}{\sum\sum h_{ij}C_{ij}D_{ij}}} \quad (5)$$

If the product item $i$ for buyers $j$ is delivered independently, the optimal cycle time of each item $i$ for buyer $j$ can be calculated from classical EOQ model.

$$\dot{t}_{ij} = \sqrt{\frac{2(S + s_{ij})}{h_{ij}C_{ij}D_{ij}}} \quad (6)$$

When the product is jointly delivered with other products, the common delivery cost $S = 0$, the optimal cycle time for each item $i$ to buyer $j$ is

$$\bar{t}_{ij} = \sqrt{\frac{2s_{ij}}{h_{ij}C_{ij}D_{ij}}} \quad (7)$$

Kaspi and Rosenblatt (1991) proposed a lower bound of $T^*$, which can be modified for MJRP to be decided by the minimum joint delivery cycle for all $\bar{t}_{ij}$,

$$T_{\min} = \underset{i,j}{Min}\left\{\bar{t}_{ij}\right\} \tag{8}$$

Goyal et al. (1993) suggested a better estimate for $T_{\min}$ in the single buyer model, which is modified by Li (2004) and employed for multiple buyers.

$$T_{\min} = \sqrt{\frac{2\left(S + \sum\sum s_{ij}/K_{ij}(t_1)\right)}{\sum\sum h_{ij}C_{ij}D_{ij}K_{ij}(t_1)}} \tag{9}$$

$$\text{where } t_1 = \underset{i,j}{Min}\left\{\bar{t}_{ij}\right\}$$

The formula (9) is obtained from the first iteration, which Kasp & Rosenblatt (1983) applied in an iterative way until the value of $K_i$ is converged in the computing procedure of JRP.

## METHODOLOGY

### Basic Concept of Genetic Algorithm

A genetic algorithm (GA) is a search technique which is used in computing to find exact or approximate solutions for optimization and search problems. Genetic algorithms (GAs) search populations of solutions for an optimization problem towards improvement by searching and selection process associated with the genes. GAs encode each possible solution into a set of genes. Each possible solution is encoded into a chromosome with binary digits or integers. Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, selection, crossover, and mutation.

The main feature of the genetic algorithm is started with a set of solutions which is represented by chromosomes and is called population. Solutions (parents) from the population are selected to mate according to their fitness and to produce a new population (offspring) by a reproductive plan. Higher fit solutions are given more opportunities to reproduce, so that offspring inherit characteristics from each parent. New generations of solutions are produced containing better genes than a typical solution in a previous generation. In the evolutionary process, the techniques of

selection, crossover and mutation operator are applied in the algorithm. If the population is not once producing offspring noticeably different from those in previous generations, the algorithm procedure is said to have converged to a set of solutions and the best solution is the optimal solution to the problem. Thus, a typical genetic algorithm requires two conditions to be defined:

- A genetic representation of the solution domain,
- A fitness function to evaluate the solution domain.

The procedure of a genetic algorithm is as follows.

1. Initialize a set of population of chromosomes with feasible solutions randomly.
2. Evaluate each chromosome in the population by a fitness function.
3. Select chromosomes from the population for reproduction.
4. Alter chromosomes in the population by applying crossover and mutation operators.
5. Evaluate the new chromosomes.
6. If the terminate condition is satisfied then return the best solution, if not, start again from point 3.

**Using $K_{ij}$ as the Genes for Chromosomes in GAK**

Khouja et al. (2000) applied the genetic algorithm (GA) to solve JRP and Chan et al. (2003) applied the GA to solve MJRP. They used the ratio of product delivery cycle time to the basic cycle time ($K_{ij}$) as the genes in a chromosome. The chromosomes represent the integer multipliers of basic cycle time ($T$) for each product delivered to each buyer. Khouja et al. (2000) used the upper bound and lower bound of $T$ from equations (5) and (8). The lower bounds on the values of $K_{ij}$ are $K_{ij}^{LB}=1$, where $i=1,2,...I.$, $j=1$. They obtained the upper bounds on the $K_{ij}$ by using equations (6) and (8). The values of $K_{ij}^{UB}$ are given by:

$$K_{ij}^{UB}=\left\lceil \frac{t_{ij}}{T_{\min}} \right\rceil \text{, where } i=1,2,...I.,\ j=1 \text{ in JRP.} \tag{10}$$

Let $u_{ij}$ denote the smallest integer such that the value of $2^{u_{ij}}$ is larger or equal to the upper bound $K_{ij}$. An integer number from the range $[1, 2^{u_{ij}}]$ can be encoded and corresponded to the binary sequence bits $u_{ij}$. The genetic representation for possible solutions is created by using the genes $K_{ij}$

$$K_{ij} = \left( \sum_{n=1}^{u_{ij}} b_n \cdot 2^{n-1} \right) + 1 \Rightarrow \rangle b_{u_{ij}} b_{u_{ij-1}} .... b_1 \langle \tag{11}$$

The integer number from the range $[1, 2^{u_{ij}}]$ is the number between the lower bound and upper bound of $K_{ij}$.

Each individual chromosome in the population represents a possible solution to the problem. The binary chromosome can be converted into a decision variables representation of $K_{ij}, \forall i \in P,\ j \in B$ after the GA operation. The evaluation function is responsible for rating these possible solutions when each decision variable is assigned. Khouja et al. (2000) used an evaluation function which can be modified for MJRP.

$$Z(K_{ij}) = \sqrt{2\left( S + \sum_{i=1}^{I} \sum_{j=1}^{J} s_{ij} / K_{ij} \right) \left( \sum_{i=1}^{I} \sum_{j=1}^{J} h_{ij} C_{ij} D_{ij} K_{ij} \right)} \tag{12}$$

Equation (12) is a function of the values $K_{ij}$, the number of decision variable $K_{ij}$ is $I \times J$. Using the genes $K_{ij}$, the total length of the binary chromosomes is $\sum_{j=1}^{J} \sum_{i=1}^{I} u_{ij}$ bits. The disadvantage of using the genes $K_{ij}$ is the total length of the chromosomes will be expanded when the problem size $I \times J$ enlarges.

**Using $T$ as the Genes for Chromosomes in GAT**

In this paper we propose the decision variable of basic cycle time $T$ as the gene to replace $K_{ij}, \forall i \in P,\ j \in B$. The number of genes $T$ in an individual chromosome is only one. The advantage of using the gene $T$ is the total length of chromosome will be NOT expanded when the problem size $I \times J$ enlargs.

The upper bound $T_{\max}$ is easily obtained from equation (5). The lower bound of $T^*$ can be obtained from equation (8) which is modified from one Kaspi and Rosenblatt (1991) proposed for solving JRP. In this paper we used another lower bound of $T^*$ from equation (9) which is modified from an equation Goyal at al (1993) suggested and tested by simulating a better estimate for $T_{\min}$ to solve JRP.

Let $l$ denote the biggest integer such that the value of $2^l$ is smaller or equal to $T_{\min}$. Let $u$ denote the smallest integer such that the value of $2^u$ is larger or equal to $T_{\max}$. A real random number from the range $[2^l, 2^u]$ can be encoded and corresponded to the binary sequence bits $v$. For computing accuracy, let

$2^v \geq 2^u * 10^m$, where $v$ is the smallest integer

Using the larger value of $m$, we obtain a higher accuracy of $T$.

The genetic representation for possible solutions using basic cycle time $T$ as the gene is created below.

$$T = \left( \left( \sum_{n=1}^{v} b_n \cdot 2^{n-1} \right) + 1 \right) \times 10^{-m} \Rightarrow \rangle b_v b_{v-1} .... b_1 \langle \tag{13}$$

The total length of our chromosomes is $v$ bits. The binary chromosome can be converted into a decision variables representation of $T$ after the GA operation.

The relationship between basic cycle time $T$ and the integer multipliers of $K_{ij}$ is discussed as follow. If a $T_x$ value is given within the bound of $[T_{\min}, T_{\max}]$, the annual common carrier cost $S/T_x$ is a constant. The objective function (2) can be described as

$$Min\ \dot{Z}(T_x, K_{ij}) = Min\ \ \dot{Z}_k(T_x, K_{ij}, \forall i \in P, j \in B) + S/T_x \tag{14}$$

where

$$\dot{Z}_k(T_x, K_{ij}, \forall i \in P, j \in B) = \sum_{i=1}^{I} \sum_{j=1}^{J} \dot{Z}_{ij}$$

$$\dot{Z}_{ij} = \left( h_{ij} C_{ij} D_{ij} K_{ij} T_x / 2 \right) + \left( s_{ij} / K_{ij} T_x \right) \tag{15}$$

where $\dot{Z}_{ij}$ is the holding cost and delivery cost of each item $i$ in buyer $j$ at $T = T_x$. The minimum value of $\dot{Z}_{ij}$ is decided by variable $K_{ij}$ in equation (15) by $\partial \dot{Z}_{ij} / \partial K_{ij} = 0$.

$\hat{K}_{ij} = \frac{1}{T_x} \sqrt{\frac{2 s_{ij}}{h_{ij} C_{ij} D_{ij}}}$, $\hat{K}_{ij}$ is the number before $K_{ij}$ rounded off, $\hat{K}_{ij} \in$ Real number.

Li (2004) modified Goyal's (1973b) round off condition in JRP and obtained the following condition for MJRP:

$K_{ij}(K_{ij} - 1) \le \frac{2 s_{ij}}{h_{ij} C_{ij} D_{ij} T^2} \le K_{ij}(K_{ij} + 1)$. Where $K_{ij}$ are integers.

Given any value $T_x$, where $T_x \in [T_{\min}, T_{\max}]$, the response value of $K_{ij}(T_x)$ from the above inequality can be obtained. The $K_{ij}$ is a function of $T_x$.

The post optimal solution of $T_x$ is $T_x^*$, which can be calculated from (4) when $K_{ij}(T_x)$ is decided.

$$T_x^* = \sqrt{\frac{2\left(S + \sum\sum s_{ij} / K_{ij}(T_x)\right)}{\sum\sum h_{ij} C_{ij} D_{ij} K_{ij}(T_x)}} \tag{16}$$

$$Z(K_{ij}(T_x^*)) = \sqrt{2\left(S + \sum_{i=1}^{I}\sum_{j=1}^{J} s_{ij} / K_{ij}(T_x^*)\right)\left(\sum_{i=1}^{I}\sum_{j=1}^{J} h_{ij} C_{ij} D_{ij} K_{ij}(T_x^*)\right)} \tag{17}$$

Equation (17) is the evaluation function of GAT in which individual chromosomes use $T$ as the gene.

## SIMULATION EXPERIMENTS AND RESULTS

The simulation experiments are designed to confirm the performance of the GAT algorithm, and then are compared with the GAK algorithm (Khouja et al. 2000) which has been tested and shown to have good performance. For comparison, the parameter ranges of test problems and the values for parameters of GA are the same with Khouja et al. 2000. The values of $D_{ij}, C_{ij}, h_{ij}$ and $s_{ij}$ for the test problem were randomly generated from uniform distribution on the ranges [100, 100 000], [1, 3], [0.5, 5.0] and [0.2, 3.0] respectively. Four different values of $I \times J$ (10, 20, 30 and 50) and four values of $S$ (5, 10, 15 and 20) were considered. The values of parameters for the GA operation are set: probability of crossover 0.6**,** the probability of mutation, denoted $P_m$**,** $P_m$=1/ (string length of chromosome), population size 30 for $I \times J$ (10, 20 and 30) and Pop 100 for $I \times J$ 50, elite=1. The above values of the parameters were selected by the best performance from testing the problems of the GAK algorithm. The termination condition is to stop computing after 500 generations or when no improved solution is obtained in 50 generations.

Table 1 Comparison between GAK and GAT solutions for total cost

| Test | Pop | $I\times J$ | $S$ | $Z_{GAT}$ better (%) | $Z_{GAT}$ better or equal (%) | Max. Cost improved (%) | Aver. Cost improved (%) |
|---|---|---|---|---|---|---|---|
| 1 | 30 | 10 | 5 | 5 | 100 | 0.2882 | 0.0036 |
| 2 | | | 10 | 1 | 100 | 0.0022 | 0.0000 |
| 3 | | | 15 | 2 | 99 | 0.0027 | 0.0000 |
| 4 | | | 20 | 2 | 99 | 0.1869 | 0.0019 |
| 5 | 30 | 20 | 5 | 27 | 94 | 0.7564 | 0.0426 |
| 6 | | | 10 | 45 | 100 | 0.7774 | 0.0353 |
| 7 | | | 15 | 36 | 99 | 0.4187 | 0.0166 |
| 8 | | | 20 | 36 | 99 | 0.3269 | 0.0116 |
| 9 | 30 | 30 | 5 | 75 | 97 | 0.4050 | 0.0158 |
| 10 | | | 10 | 77 | 98 | 0.6672 | 0.0378 |
| 11 | | | 15 | 87 | 100 | 0.5040 | 0.0336 |
| 12 | | | 20 | 70 | 100 | 0.8406 | 0.0464 |
| 13 | 100 | 50 | 5 | 98 | 99 | 0.6273 | 0.0423 |
| 14 | | | 10 | 100 | 100 | 0.5975 | 0.1342 |
| 15 | | | 15 | 100 | 100 | 0.4827 | 0.0937 |
| 16 | | | 20 | 100 | 100 | 0.5150 | 0.0963 |
| Average | | | | 54 | 99 | 0.4624 | 0.0382 |

Note: '$Z_{GAT}$ better (or equal)' means that comparing the effectiveness of the solution , the results of percentage by GAT algorithm is better (or equal) than GAK method.

For each combination of $I\times J$ and $S$, 100 problems were generated and solved by using the GAK and GAT for a total 1600 problems. A value of $m=8$ was used to compute the bits $v$(length of chromosomes) with GAT for solution accuracy. The simulation results are summarized in Tables 1 and 2. In Table 1, the label of '$Z_{GAT}$ better' means that the solution of objective function $Z$ minimized by GAT is better than by GAK, and the label of '$Z_{GAT}$ better or equal' means that the solution of objective function $Z$ minimized by GAT is better than or is equal to GAK. The percentage of all test problems of '$Z_{GAT}$ better' is 54% and '$Z_{GAT}$ better or equal' is 99%. The maximum saving cost in the problem of Test 12 reaches 0.8406 % of the total cost. The average of maximum saving cost is 0.4624 %. For the large size problem ($I\times J$=50), the performance of solutions quality with the GAT is much better than with the GAK. As shown in Table 1, $I\times J$=50, '$Z_{GAT}$ better' is 98% for $S$=5 and 100% for $S$=10, 15 and 20. And comparing the termination generations in Table 2, the average termination generations with GAT are 52.85 and with GAK are 500. Obviously the GAT convergence to a good solution is much faster than with GAK.

The average terminated generations with GAK was improved 89.43% by using GAT. Comparing the CPU running time, the average CPU time for each problem with GAT is 0.786 seconds and with GAK it is 44.011 seconds. The average computing time with GAK can be improved 97.805% by the proposed GAT. Eventually, GAT is superior to GAK both in the quality of solutions as CPU running time.

Table 2 Comparison between GAK and GAT solutions for # Gen. and CPU time

| Test | Pop | $I\times J$ | $S$ | Average terminated Gen. | | | Average CPU time (sec) | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | GAT | GAK | Improved % | GAT | GAK | Improved % |
| 1 | 30 | 10 | 5 | 51.03 | 500 | 89.79 | 0.668 | 18.203 | 96.328 |
| 2 | | | 10 | 51.20 | 500 | 89.76 | 0.654 | 17.991 | 96.361 |
| 3 | | | 15 | 51.08 | 500 | 89.78 | 0.652 | 17.960 | 96.367 |
| 4 | | | 20 | 51.22 | 500 | 89.75 | 0.654 | 17.979 | 96.361 |
| 5 | 30 | 20 | 5 | 51.90 | 500 | 89.62 | 0.714 | 32.779 | 97.821 |
| 6 | | | 10 | 51.53 | 500 | 89.69 | 0.710 | 32.798 | 97.834 |
| 7 | | | 15 | 52.56 | 500 | 89.49 | 0.726 | 32.842 | 97.788 |
| 8 | | | 20 | 52.17 | 500 | 89.56 | 0.717 | 32.717 | 97.810 |
| 9 | 30 | 30 | 5 | 53.87 | 500 | 89.22 | 0.812 | 47.717 | 98.297 |
| 10 | | | 10 | 52.29 | 500 | 89.54 | 0.788 | 47.686 | 98.297 |
| 11 | | | 15 | 53.19 | 500 | 89.36 | 0.803 | 47.747 | 98.317 |
| 12 | | | 20 | 53.23 | 500 | 89.35 | 0.803 | 47.693 | 98.315 |
| 13 | 100 | 50 | 5 | 54.81 | 500 | 89.03 | 0.960 | 77.522 | 98.755 |
| 14 | | | 10 | 54.48 | 500 | 89.10 | 0.959 | 77.523 | 98.762 |
| 15 | | | 15 | 55.69 | 500 | 88.86 | 0.982 | 77.492 | 98.732 |
| 16 | | | 20 | 55.40 | 500 | 88.92 | 0.976 | 77.524 | 98.740 |
| Average | | | | 52.85 | 500 | 89.43 | 0.786 | 44.011 | 97.805 |

Note: Average terminated Gen. means the average terminated generation for each tested problem.
Average CPU time means the average CPU running time for each tested problem.

## CONCLUSIONS

In this paper, we propose using a New Gene $T$ to replace the genes $K_{ij}$ for genetic algorithm to solve MJRP. The major advantage of GAT with New Gene $T$ is that the number of genes always is kept to only one in the chromosomes and the length of chromosomes is not expanded when the problem size enlarges. The New Gene shortens the CPU running time of GAT, which is important from a practical point of view. It can be used to solve the generalized MJRP efficiently. GAT outperformed GAK not only reducing CPU running time but also improving the

quality of the solutions. Finally, we would like to remark that the generalized MJRP model is not always realistic, and it may be further researched to handle constrained problem.

## REFERENCES

Arkin, E., Joneja, D., and Roundy, R. (1989). Computational complexity of incapacitated multi-echelon production planning problems. *Operation Research, 8(2),* 61-66.

Chan, C. K., Cheung, B. K-S., and Langevin, A. (2003). Solving the multi-buyer joint replenishment problem with a modified genetic algorithm. *Transportation Research Part B: Methodological, 37(3),* 291-299.

Chan, C. K., Leon L. Y., Chi T. N., Cheung, B., and Langevin, A. (2006). Scheduling of multi-buyer joint replenishments. *International Journal of Production Economics, 102(1),* 132-142.

Frenk, J. B. G., Kleijn, M. J., and Dekker, R. (1999). An efficient algorithm for a generalized joint replenishment problem. *European Journal of Operational Research, 118(2),* 413-428.

Goyal, S. K. (1973b). Determination of economic packaging frequency for items jointly replenished. *Management Science, 20(2),* 232-235.

Goyal, S. K. (1974). Determination of optimum packaging frequency of items jointly replenished. *Management Science, 21(4),* 436-443.

Goyal, S. K., and Deshmukh, S. G. (1993). A note on 'The economic ordering quantity for jointly replenishing items'. *International Journal of Production Research, 31(12),* 2959-2961.

Hoque, M. A. (2006). An optimal solution technique for the joint replenishment problem with storage and transport capacities and budget constraints. *European Journal of Operational Research, 175(2),* 1033-1042.

Kaspi, M. and Rosenblatt, M. J. (1983). An improvement of Siliver's algorithm for the joint replenishment problem. *IIE Transactions, 15(3),* 264-267.

Kaspi, M., and Rosenblatt, M. J. (1991). On the economic ordering quantity for jointly replenished items. *International Journal of Production Research, 29(1),* 107-114.

Khouja, M., Michalewicz, Z., and Sandeep, S. S. (2000). A comparison between genetic algorithms and the RAND method for solving the joint replenishment problem. *Production Planning & Control, 11(6),* 556-564.

Li, Q. (2004). Solving the multi-buyer joint replenishment problem with the RAND method. *Computers and Industrial Engineering*, *46(4)*, 755-762.

Moon, I. K., and Cha, B. C. (2006). The joint replenishment problem with resource restriction. *European Journal of Operational Research*, *173(1)*, 190-198.

Olsen, A. L. (2005). An evolutionary algorithm to solve the joint replenishment problem using direct grouping. *Computers & Industrial Engineering*, *48(2)*, 223-235.

Wildeman, R. E., Frenk, J. B. G., and Dekker, R. (1997). An efficient optimal solution method for the joint replenishment problem. *European Journal of Operational Research*, *99(2)*, 433-444.