Available online at www.sciencedirect.com
ScienceDirect
Transportation Research Part E 44 (2008) 720–730
TRANSPORTATION RESEARCH PART E
www.elsevier.com/locate/tre

The joint replenishment and delivery scheduling of the one-warehouse, *n*-retailer system

B.C. Cha [a], I.K. Moon [b,*], J.H. Park [a]

[a] *Postal Technology Research Center, Electronics and Telecommunications Research Institute, Daejeon 305-700, Republic of Korea*
[b] *Department of Industrial Engineering, Pusan National University, Busan 609-735, Republic of Korea*

Received 3 August 2006; received in revised form 5 March 2007; accepted 16 May 2007

Abstract

We deal with the joint replenishment and delivery scheduling of the one-warehouse, *n*-retailer system in this paper. We suggest a more flexible policy for the joint replenishment and delivery scheduling of a warehouse compared with the existing researches. We introduce the mathematical model and two efficient algorithms for the joint replenishment and delivery scheduling of the warehouse. Subsequently, we develop the hybrid genetic algorithm (GA) and compare it with two efficient heuristic algorithms for extensive computational experiments. Further, we show the advantages of our GA in dealing easily with resource restrictions.

Keywords: Joint replenishment; One-warehouse *n*-retailer system; Genetic algorithm

1. Introduction

Supply chain management (SCM) refers to the management of material and information flow both in and between facilities, such as retailers, warehouses/distribution centers, and suppliers. SCM is an area that has recently received a great deal of attention in the business community. In recent years, many companies have realized that considerable cost savings can be achieved by integrating inventory control and delivery policies throughout their supply chains.

In this paper, we focus on a two-stage supply chain comprising one warehouse and *n*-retailers. This one-warehouse, *n*-retailer problem has received considerable attention from researchers (Silver et al., 1998). Schwarz (1973) determined the properties of an optimal solution to the one-warehouse, *n*-retailer problem and provided lower bounds on the average cost of the optimal policy. He showed that an optimal policy can be very complex in form; in particular, it requires that the order quantity at one or more of the locations vary with time despite all the relevant demand and cost factors being time-invariant. Maxwell and Muckstadt

* Corresponding author. Tel.: +82 51 510 2451; fax: +82 51 512 7603.
E-mail address: ikmoon@pusan.ac.kr (I.K. Moon).

1366-5545/$ - see front matter
doi:10.1016/j.tre.2007.05.010

(1985) restricted their policy to the stationary-nested one. A policy is considered as stationary if each facility always orders the same quantity at equally spaced points in time. A nested policy is one wherein each time that any stage orders, all of its successors follow suit and order. Roundy (1985) demonstrated that nested policies can have rather low effectiveness in the worst case, and he introduced two simple policies: q-optimal integer-ratio and optimal power-of-two policies. He proved that for any data set, the effectiveness of q-optimal integer-ratio and optimal power-of-two policies is at least 94% and 98%, respectively. Roundy (1986) extended his study to the multi-item problem. Viswanathan and Mathur (1997) presented a new heuristic algorithm for a stationary-nested joint replenishment policy. They incorporated the vehicle routing problem to the multi-item one-warehouse, n-retailer one. Recently, Abdul-Jalbar et al. (2003) compared a centralized and a decentralized case for the one-warehouse, n-retailer problem. In the latter case, retailers make decisions independently. In the former case, they proposed two heuristic algorithms, one considering a common replenishment time, and the other different reorder times. Hong and Park (2006) presented a mixed integer program for the vendor-managed inventory policy with vehicle routing problems which consists of a single supplier and multiple retailers. Eum et al. (2005) developed a neural network algorithm to decide the allocation policy for the producer which has multiple retailers.

Graves (1979) showed that the joint replenishment problem (JRP) is closely related to the one-warehouse, n-retailer problem. During the last three decades following the early work of Shu (1971), the JRP has attracted the attention of many researchers. Goyal (1974) proposed an enumeration approach to obtain an optimal solution. Van Eijs (1993) proposed a similar optimal algorithm suggested by Goyal (1974). Viswanathan (1996, 2002) proposed a fast algorithm obtaining the optimal cyclic policy for the JRP compared with those of Goyal (1974) and Van Eijs (1993). Silver (1975, 1976) discussed the advantages and disadvantages of coordinating replenishments and presented an extremely simple non-iterative procedure for solving the JRP. Kaspi and Rosenblatt (1991) proposed an approach based on trying several values of the basic cycle time between the minimum and maximum values. Then, they applied the heuristic of Kaspi and Rosenblatt (1983) to each value of the basic cycle time, which is a modified version of the algorithm of Silver (1975). They demonstrated that their procedure (known as the RAND method) outperforms all the available heuristics. Later, Goyal and Deshmukh (1993) proposed an improvement of the lower bound used by Kaspi and Rosenblatt (1991). Wildeman et al. (1997) presented a new optimal solution approach based on Lipschitz optimization to obtain a solution with an arbitrarily small deviation from the optimal value. Khouja et al. (2000) applied genetic algorithms (GAs) to the JRP and compared the performance of their GA with the heuristic algorithm of Kaspi and Rosenblatt (1991). Recently, for the multi-buyer joint replenishment problem, Chan et al. (2003) proposed a modified genetic algorithm and Li (2004) developed an efficient RAND method.

However, the joint replenishment of multi-items was not considered in the above one-warehouse, n-retailer models discussed above. Our objective is to investigate the effectiveness of the joint replenishment policies of a warehouse as shown in Fig. 1.

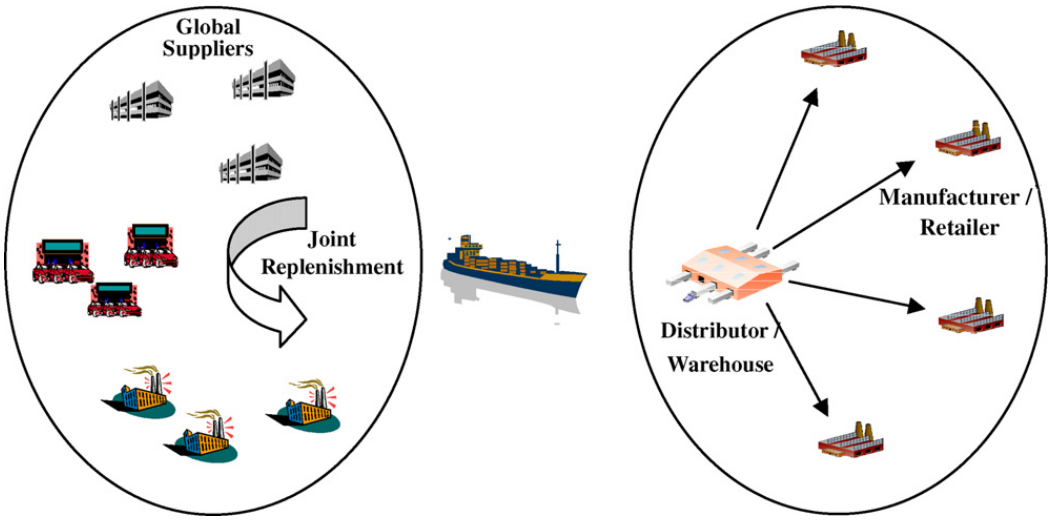

Fig. 1. Joint replenishment and delivery scheduling of a warehouse.

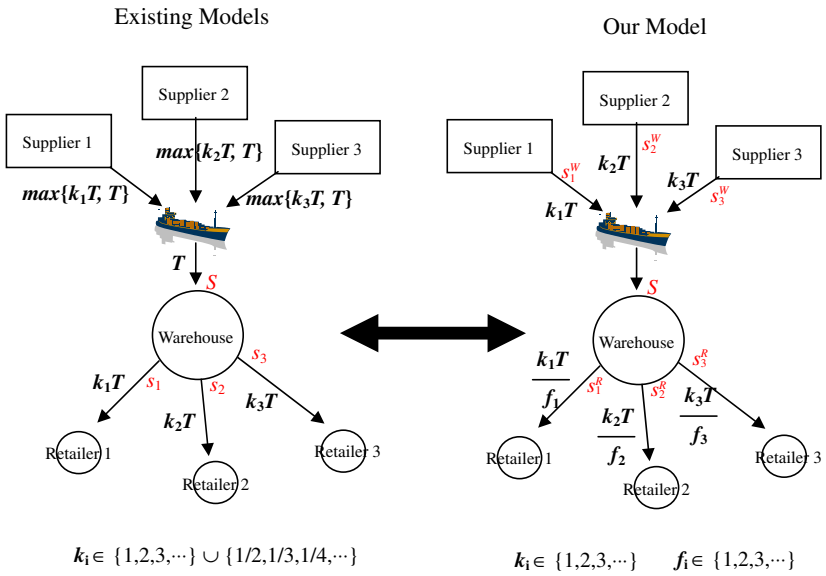

Fig. 2. Comparison between existing models and our model.

The warehouse delivers to an individual manufacturer or retailer after it replenishes jointly from the suppliers by taking into account the demand for each item at the manufacturer or retailer. The warehouse or distribution center that has many associated manufacturers or retailers can decrease the logistics cost significantly by replenishing jointly. The reduction effects will be higher in the case of the high procurement cost incurred from importing materials or parts from overseas. In this paper, we will introduce a more flexible joint replenishment policy for the warehouse (or third-party logistics company) that procures multiple items for multiple manufacturers or retailers.

Following the assumptions of Lu and Posner (1994), we consider n different types of items and assume that all items can be stored at the warehouse. Further, we assume that item i is stocked and sold only by retailer i. The warehouse replenishes item i at every integer multiple (k_i) of the basic cycle time (T) and delivers it to retailer i. Lu and Posner (1994) provided two heuristic algorithms for the one-warehouse, n-retailer problem and mentioned that the problem can be extended to the JRP. Our model is more flexible than those of Lu and Posner (1994) in terms of the joint replenishment to the suppliers and individual distribution policies to the retailers as shown in Fig. 2. For example, the policy for $k_1 = 2$ and $f_1 = 2$ could not be represented in the existing models. Moreover, the minor cost for individual item has not been considered in the joint replenishment of the warehouse in the existing models.

Section 2 introduces the mathematical model and two different heuristic algorithms for the joint replenishment and delivery scheduling problem. We develop a hybrid GA in Section 3. In Section 4, we perform computational experiments in order to analyze the effectiveness of our GA. Further, in Section 5, we show the extension ability of our GA in dealing with resource restrictions. Finally, we summarize the conclusions of the present work.

2. Joint replenishment and delivery scheduling

In order to discuss the joint replenishment and delivery scheduling problem in the one-warehouse, n-retailer problem, the following notations are defined:

i	index of item, $i = 1, 2, \ldots, n$
D_i	demand rate of item i
S^{W}	warehouse's major ordering cost
s_i^{W}	warehouse's minor ordering cost of item i
h_i^{W}	warehouse's inventory holding cost of item i per unit per unit time

s_i^R warehouse's outbound transportation cost of item i
h_i^R retailer's inventory holding cost of item i per unit per unit time
T warehouse's basic cycle time (decision variable)
k_i integer number that decides the replenishment schedule of item i (decision variable)
f_i integer number that decides the outbound schedule of item i (decision variable)

The stationary policy is that a warehouse delivers item i to retailer i at the same time interval as shown in Fig. 3 (where the order quantity does not change with time). The warehouse replenishes item i at every integer multiple k_i of the basic cycle time T. The total cost function is composed of the sum of the ordering (major and minor) cost, inventory holding cost, and outbound transportation cost of a warehouse and as well as the total of the inventory holding costs of retailers.

According to the above definition, the total relevant cost per unit time to be minimized is given by

$$TC(T,k_i's,f_i's)=\frac{S^W+\sum_{i=1}^{n}\frac{s_i^W}{k_i}}{T}+\sum_{i=1}^{n}\frac{(f_i-1)k_iTD_ih_i^W}{2f_i}+\sum_{i=1}^{n}\frac{f_is_i^R}{k_iT}+\sum_{i=1}^{n}\frac{k_iTD_ih_i^R}{2f_i} \tag{1}$$

To find the T, k_is, and f_is that minimize the total relevant cost per unit time, Cha and Moon (2004) used the following optimality condition of each decision variable.

For a given set of k_is and f_is, the optimal basic cycle time T can be easily obtained from the first order derivative of the total cost function since the total cost function is convex in T. T is obtained from Eq. (2).

$$T^*=\sqrt{\frac{2\left(S^W+\sum_{i=1}^{n}\frac{s_i^W+f_is_i^R}{k_i}\right)}{\sum_{i=1}^{n}k_iD_i\left(h_i^W+\frac{h_i^R-h_i^W}{f_i}\right)}} \tag{2}$$

For a given set of f_is and T, we can derive the optimality condition of k_i from the following two conditions:

$$TC(k_i)\leqslant TC(k_i+1)\quad\text{and}\quad TC(k_i)\leqslant TC(k_i-1)$$

Therefore, the optimality condition of k_i is defined as Eq. (3).

$$k_i(k_i-1)\leqslant\frac{2(s_i^W+f_is_i^R)}{T^2D_i\left(h_i^W+\frac{h_i^R-h_i^W}{f_i}\right)}\leqslant k_i(k_i+1) \tag{3}$$

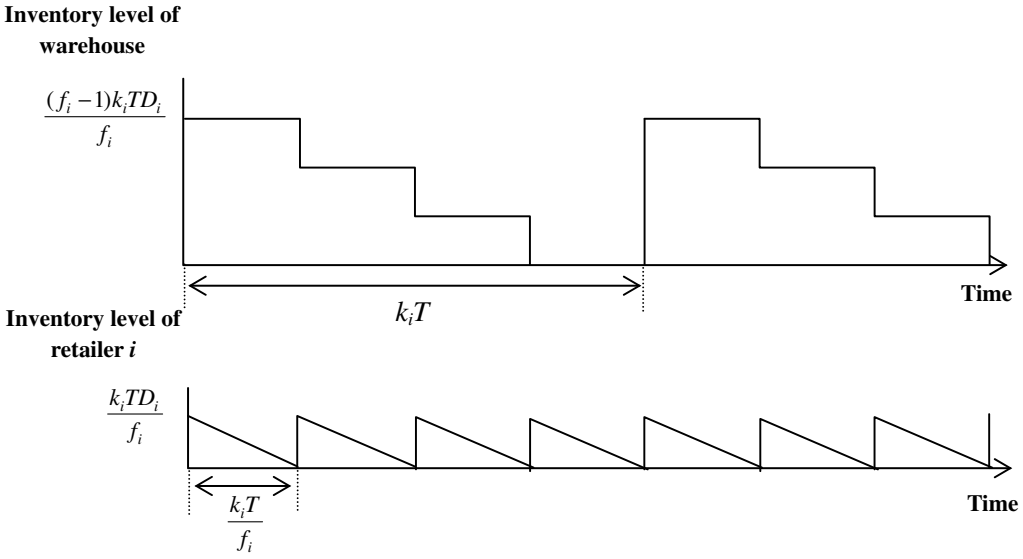

Fig. 3. Replenishment and delivery cycle of item i for the stationary policy.

For a given set of k_is and T, we can derive the optimality condition of f_i from the following two conditions:

$$TC(f_i) \leqslant TC(f_i + 1) \quad \text{and} \quad TC(f_i) \leqslant TC(f_i - 1)$$

The optimality condition of f_i is defined as Eq. (4).

$$f_i(f_i - 1) \leqslant \frac{k_i^2 T^2 D_i (h_i^R - h_i^W)}{2 s_i^R} \leqslant f_i(f_i + 1) \tag{4}$$

If $h_i^W \geqslant h_i^R$, $f_i = 1$ since the total cost function is an increasing step function in f_i.

Using the above optimality conditions, we develop two efficient algorithms. One is a simple recursive algorithm, and the other is a modified RAND algorithm that adopts the research idea of Kaspi and Rosenblatt (1991). The procedure of each heuristic algorithm is as follows:

Joint replenishment and delivery – Simple heuristic (JRD–SH)

Step 1. Set the iteration number $r = 0$.
Put $(k_1(r), k_2(r), \ldots, k_n(r)) = 1$, $(f_1(r), f_2(r), \ldots, f_n(r)) = 1$ and $T(r) = 0$. Go to Step 2.
Step 2. Set $r = r + 1$, and go to Step 3.
Step 3. For a given set of $k_i(r-1)$s and $f_i(r-1)$s, find the optimal value of T using Eq. (2).
Set $T(r) = T$. If $T(r) = T(r-1)$, stop. Otherwise, go to Step 4.
Step 4. For a given value of $T(r)$ and a given set of $f_i(r-1)$s, find the optimal values of k_i using Eq. (3). Set $k_i(r) = k_i$, and go to Step 5.
Step 5. For a given value of $T(r)$ and a given set of $k_i(r)$s, find the optimal values of f_i using Eq. (4). Set $f_i(r) = f_i$, and go to Step 2.

Joint replenishment and delivery – RAND algorithm (JRD–RAND)

Step 1. Compute $T_{\max} = \sqrt{2(S + \sum_{i=1}^{n} s_i)/\sum_{i=1}^{n} D_i h_i}$ and $T_{\min} = \min \sqrt{2s_i/D_i h_i}$ for each i.
Step 2. Divide the range $[T_{\min}, T_{\max}]$ into m different equally spaced values of $T(T_1, \ldots, T_j, \ldots, T_m)$. The value of m is to be decided by the decision maker. Set $j = 0$.
Step 3. Set $j = j + 1$ and $r = 0$. Put $T_j(r) = T_j$ and $(f_1(r), f_2(r), \ldots, f_n(r)) = 1$.
Step 4. Set $r = r + 1$.
Step 5. For a given value of $T_j(r-1)$ and a given set of $f_i(r-1)$s, find the optimal values of k_i using Eq. (3). Put $k_i(r) = k_i$.
Step 6. For a given value of $T_j(r-1)$ and a given set of $k_i(r)$s, find the optimal values of f_i using Eq. (4). Put $f_i(r) = f_i$.
Step 7. For a given set of $k_i(r)$s and $f_i(r)$s, find the optimal value of T using Eq. (2). Put $T_j(r) = T$.
Step 8. If $T_j(r) \neq T_j(r-1)$, go to Step 4.
Otherwise, put $T_j^* = T_j(r)$, $k_{ij}^* = k_i^*(r)$, and $f_{ij}^* = f_i^*(r)$.
Compute TC_j for this $\left(T_j^*, k_{ij}^* s, f_{ij}^* s\right)$.
Step 9. If $j \neq m$, go to Step 3.
Otherwise, stop and select $(T_j^*, k_{ij}^* s, f_{ij}^* s)$ with the minimum TC.

3. Hybrid GA

In this section we present a new GA approach for solving the JRP of the one-warehouse, n-retailer system. The main ideas of the GA have been introduced, and we will demonstrate how we apply this GA to our problem. The GA introduced by Khouja et al. (2000) is suitable for solving the JRP that has the important feature that it can be formulated as a problem having one continuous decision variable (basic cycle T) and a set of integer decision variables (n integer multiples k_i of a basic cycle T).

GAs, which have been widely used to solve operations management problems during the last decade (Aytug et al., 2003), are stochastic search algorithms based on the mechanism of natural selection and natural genetics. Unlike conventional search techniques, GAs start with an initial set of (random) solutions known as a

population. Each individual in the population is called a chromosome that represents a solution to the problem at hand. The chromosomes evolve through successive iterations, known as generations. During each generation, the chromosomes are evaluated by using some measures of fitness. In general, the GA is applied to spaces that are too large to be exhaustively searched. It is generally accepted that any GA that is used to solve a problem must have basic components; however, it can have different characteristics depending on the problem under study.

We explain our overall strategies including chromosome style based on the following:

- Representation and initialization.
- Objective and fitness functions.
- Reproduction, crossover, and mutation.

3.1. Representation and initialization

The appropriate representation of a solution plays a key role in the development of a GA. Unlike the method for solving a general JRP that is presented in Khouja et al. (2000), solving the JRP of the one-warehouse, n-retailer system involves the determination of the basic cycle T, the replenishment schedule variables (k_is), and the outbound schedule variables (f_is). In our GA, we can search k_is and f_is through the operations of the GA and determine the basic cycle T through the optimality condition of T for the given k_is and f_is.

As shown in Fig. 4, our decoding chromosome is composed of two parts. One is for the replenishment schedule, and the other for the outbound schedule of each item. The ith gene of the first part of the chromosome indicates the integer value of k_i that decides the replenishment schedule of item i. The ith gene of the second part of the chromosome indicates the integer value of f_i that decides the outbound schedule of item i. Therefore, we can determine the values of k_i and f_i by using this chromosome. Our chromosome requires only $2n$ genes to decide the replenishment schedule variables (k_is) and the outbound schedule variables (f_is). In our study, we use a random number representation for the following reasons:

(i) Our GA is always searching for a suitable feasible region without being influenced by any crossover and mutation.
(ii) It is very easy to decode a chromosome expressed by a random number representation to a feasible solution.

3.2. Objective and fitness function

In this subsection, we will demonstrate how our chromosome can be decoded to a feasible solution. Each chromosome of the population is evaluated by the following steps:

Step 1. Each chromosome is decoded to a feasible solution (k_is, f_is).
Step 2. The optimal basic cycle T is determined for a given (k_is, f_is) from Eq. (2).
Step 3. The total relevant cost TC is computed for a given (T, k_is, and f_is).

Our decoding process for each gene of the first part of our chromosome is as follows:

$$k_i = k_i^{\mathrm{LB}} + \left\lfloor \left(k_i^{\mathrm{UB}} - k_i^{\mathrm{LB}} + 1\right) \times \mathrm{Gene}(i) \right\rfloor$$

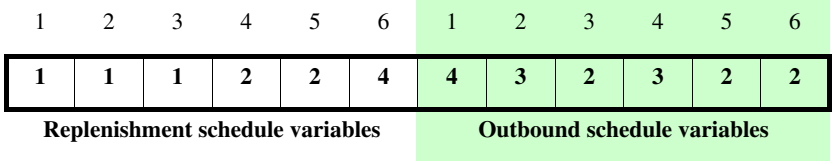

Fig. 4. Decoding chromosome.

$\lfloor G \rfloor$ is the function that finds the integer number that is less than or equal to G. For each gene of the second part of our chromosome, the decoding process is as follows:

$$f_i = f_i^{\mathrm{LB}} + \left\lfloor \left(f_i^{\mathrm{UB}} - f_i^{\mathrm{LB}} + 1\right) \times \mathrm{Gene}(i) \right\rfloor$$

By setting a suitable range of k_is and f_is, we can define the solution space including the optimal solution. It is evident that considerably smaller k_is and f_is lead to a better solution space, as far as it contains the optimal solution.For solving the general JRP, Khouja et al. (2000) used $\left(k_i^{\mathrm{LB}} = 1, k_i^{\mathrm{UB}} = \left\lceil T_i^{\mathrm{IN}}/T_{\min} \right\rceil\right)$, where $T_i^{\mathrm{IN}} = \sqrt{2(S+s_i)/D_i h_i}$ is the individual optimal cycle time for product i, which is obtained from the EOQ model. Moon and Cha (2006) defined tighter lower and upper bounds of k_i from the well-known optimality condition (Goyal, 1973):

$$k_i(k_i - 1) \leqslant \frac{2s_i}{D_i h_i T^2} \leqslant k_i(k_i + 1)$$

In other words, they used the lower and upper bounds of k_i from the following two equations:

$$k_i^{\mathrm{LB}}(k_i^{\mathrm{LB}} - 1) \leqslant \frac{2s_i}{D_i h_i T_{\max}^2} \leqslant k_i^{\mathrm{LB}}(k_i^{\mathrm{LB}} + 1)$$

$$k_i^{\mathrm{UB}}(k_i^{\mathrm{UB}} - 1) \leqslant \frac{2s_i}{D_i h_i T_{\min}^2} \leqslant k_i^{\mathrm{UB}}(k_i^{\mathrm{UB}} + 1)$$

where $T_{\max} = \sqrt{2\left(S + \sum_{i=1}^{n} s_i\right)/\sum_{i=1}^{n} D_i h_i}$ and $T_{\min} = \min\left(\sqrt{2s_i/D_i h_i}\right)$.

However, in this problem, it is difficult to define the lower and upper bounds of the variables k_i and f_i since the two variables influence each other. Therefore, we use $k_i^{\mathrm{LB}} = 1$ and $f_i^{\mathrm{LB}} = 1$, which are clearly the lower bounds of item i. To obtain the upper bounds of k_i and f_i, we use values of one and a half times the optimal values obtained from JRD–RAND.

3.3. Reproduction, crossover and mutation

Various evolutionary methods can be applied to this problem. We use the $(\mu + \lambda)$ selection for selecting individuals for reproduction. With this strategy, μ parents and λ offsprings compete for survival, and the μ best of the offsprings and parents are selected as parents of the next generation. Further, we produce offsprings through a one-point crossover. Whenever an offspring is produced, mutation is applied with probability P_{m}. The operation of mutation replaces one gene of the chromosome that is chosen at random with a new random number between (0,1).

We use a numerical example to compare the three algorithms. The data and results of the example are given in Tables 1 and 2. Both JRD–RAND and genetic algorithm found an optimal solution.

Table 1
Data for the example ($S^{\mathrm{W}} = \$200$)

Item i	1	2	3	4	5	6
D_i	10,000	5000	3000	1000	600	200
s_i^{W}	45	46	47	44	45	47
h_i^{W}	1	1	1	1	1	1
s_i^{R}	5	5	5	5	5	5
h_i^{R}	1.5	1.5	1.5	1.5	1.5	1.5

Table 2
Comparison between two heuristic algorithms and GA

Algorithm	T	k_i	f_i	TC	%
JRD–SH	0.1973	1, 1, 1, 1, 2, 3	4, 3, 2, 1, 2, 2	\$4850.39	0.45
JRD–RAND	0.1881	1, 1, 1, 2, 2, 4	4, 3, 2, 3, 2, 2	\$4828.89	–
GA	0.1881	1, 1, 1, 2, 2, 4	4, 3, 2, 3, 2, 2	\$4828.89	–

4. Computational experiments

In this section we compare the performances of three algorithms for a number of randomly generated problems. The problem data are generated from a uniform distribution of the ranges presented in Table 3.

To determine the appropriate values of the GA parameters, we conducted experiments on 30 randomly generated problems ($S = 100$, $n = 50$). Four different probabilities of crossover (0.4, 0.6, 0.8 and 1.0) and four different probabilities of mutation (0.05, 0.10, 0.15 and 0.20) are considered. The termination condition is to stop if no improvement is made in 100 generations. Computational results for the test problems are shown in Table 4.

Table 4 shows that the mean error is best when the probabilities of crossover and mutation are 0.8 and 0.1, respectively. Therefore, we determine the probabilities of crossover and mutation to be 0.8 and 0.1, respectively.

Four different values of n (10, 20, 30 and 50) and four different values of S (100, 200, 300 and 400) are considered. For $n = 10$, three algorithms are compared with the optimal solution from full enumeration. We compare them with their best solution for other values of n since it took too much time to find an optimal solution using full enumeration. For each combination of n and S, 100 problems are generated and solved using the JRD–SH, GA, and JRD–RAND for a total of 1600 problems. A value of $m = n$ is used in the JRD–RAND. In the GA, the population size of $2n$ is used. A summary of the computational results for 1600 randomly generated problems is shown in Tables 5 and 6.

As shown in Tables 5 and 6, the JRD–RAND is superior to the other two algorithms. However, Table 6 also shows that the performance of the GA is extremely good as compared with the best solution of each problem (the maximum error found is 0.3660% and the average error found is only 0.0268%). In addition, the GA can be easily expanded to more complex problems with several constraints while it would be difficult to extend the JRD–RAND.

Table 3
The ranges of parameters

D_i	s_i^W	s_i^R	h_i^W	h_i^R
[500, 5000]	[30, 50]	$[0.1s_i^W, 0.3s_i^W]$	[0.5, 3.0]	$[1.2h_i^W, 2.0h_i^W]$

Table 4
Computational results for test problems

P_m/P_c	Mean errors (%)				Mean generation numbers			
	0.4	0.6	0.8	1.0	0.4	0.6	0.8	1.0
0.05	0.0569	0.0964	0.0677	0.0470	471	421	413	365
0.10	0.0450	0.0518	0.0413	0.0416	399	351	333	296
0.15	0.0496	0.0568	0.0480	0.0417	381	296	280	265
0.20	0.0804	0.0626	0.0515	0.0474	336	292	257	244

Table 5
Comparison between three algorithms and the optimal solution (% error)

n	S	Number of finding an optimal solution			JRD–SH		GA		JRD–RAND	
		JRD–SH	GA	JRD–RAND	Max.	Avg.	Max.	Avg.	Max.	Avg.
10	100	58	81	93	1.1713	0.0907	0.4208	0.0230	0.0603	0.0018
	200	67	82	94	0.6467	0.0463	0.6088	0.0188	0.0119	0.0002
	300	74	68	97	0.3997	0.0302	0.3255	0.0228	0.0644	0.0007
	400	77	75	92	0.4308	0.0230	0.4102	0.0273	0.0384	0.0006
Max.					1.1713		0.6088		0.0644	
Avg.		69	77	94		0.0475		0.0230		0.0222

Table 6
Comparison of three algorithms (% error)

n	S	Number of finding the best solution			JRD–SH		GA		JRD–RAND	
		JRD–SH	GA	JRD–RAND	Max.	Avg.	Max.	Avg.	Max.	Avg.
20	100	29	63	96	0.8601	0.1789	0.2246	0.0306	0.0159	0.0002
	200	38	65	97	0.6776	0.0747	0.3248	0.0291	0.0689	0.0010
	300	51	77	97	0.5185	0.0408	0.2038	0.0112	0.0031	0.0001
	400	60	66	95	0.3060	0.0226	0.1964	0.0159	0.0485	0.0008
30	100	7	48	86	1.0772	0.2673	0.3660	0.0379	0.0880	0.0035
	200	13	58	92	0.6656	0.1269	0.2823	0.0194	0.0771	0.0012
	300	26	57	93	0.3777	0.0753	0.2606	0.0269	0.0125	0.0004
	400	35	63	95	0.2602	0.0455	0.1512	0.0179	0.0333	0.0006
50	100	1	38	87	1.1011	0.3661	0.3299	0.0507	0.0832	0.0025
	200	2	34	87	0.8082	0.2344	0.2183	0.0311	0.0589	0.0023
	300	8	42	92	0.4663	0.1551	0.1980	0.0284	0.0455	0.0017
	400	5	45	87	0.3389	0.1095	0.1496	0.0221	0.0352	0.0018
Max.					1.1011		0.3660		0.0880	
Avg.		23	55	92		0.1414		0.0268		0.0013

5. Resource restrictions

Most real life JRPs exist under conditions of limited resources such as storage, transport equipment capacity, and budget. Goyal (1975) considered a JRP with one resource constraint and developed a heuristic algorithm using the Lagrangian multiplier. Khouja et al. (2000) introduced the advantage of GA in handling constrained JRPs. Moon and Cha (2006) compared their GA with Goyal's algorithm and showed the extension ability of their GA to easily deal with many constraints.

It is difficult for recursive heuristic algorithms to deal with many constraints. However, we can easily extend our GA to solve the problems with resource restrictions. The following two types of transportation restrictions can be considered. One is the joint replenishment restriction, and the other the outbound delivery restriction. The former can be represented as follows:

$$\sum_{i=1}^{n} D_i k_i T b_i \leqslant B^{\mathrm{W}} \tag{5}$$

where b_i is the unit weight of item i and B^{W} is the maximum weight capacity of a full-ship load for the joint replenishment of all items. Similarly, the latter can be represented as follows:

$$\frac{D_i k_i T b_i}{f_i} \leqslant B_i^{\mathrm{R}} \tag{6}$$

where B_i^{R} is the maximum weight capacity of a full truck load for the outbound delivery of item i.

We can easily apply our GA to these types of problems by only changing Eq. (2) to find the optimal T that satisfies the resource restriction.

Proposition 1. *For a given set of k_is and f_is, the optimal basic cycle time T is $T^* = \min(T^0, T^1)$ where*

$$T^0 = \sqrt{\frac{2\left(S^{\mathrm{W}} + \sum_{i=1}^{n} \frac{s_i^{\mathrm{W}} + f_i s_i^{\mathrm{R}}}{k_i}\right)}{\sum_{i=1}^{n} k_i D_i \left(h_i^{\mathrm{W}} + \frac{h_i^{\mathrm{R}} - h_i^{\mathrm{W}}}{f_i}\right)}}$$

and

$$T^1 = \min\left(\frac{B^{\mathrm{W}}}{\sum_{i=1}^{n} D_i k_i b_i}, \frac{f_i B_i^R}{D_i k_i b_i}\right)$$

Table 7
Result for the resource restriction

	k_i	f_i	T^*	TC
No. Restriction	1, 1, 1, 2, 2, 4	4, 3, 2, 3, 2, 2	0.1881	\$4828.89
Resource restriction	1, 1, 1, 2, 2, 4	6, 3, 2, 3, 2, 2	0.1818	\$4843.83

for all i. T^1 *is obtained from the restriction of both Eqs.* (5) *and* (6).

Proof. If $T^0 \leqslant T^1$, $TC(T)$ is a convex function in the interval $[0,T^1]$. In this case, the optimal basic cycle time $T^* = T^0$. Otherwise, $T^0 \geqslant T^1$, as $TC(T)$ is a decreasing function in the interval $[0,T^1]$. It is clear that the optimal basic cycle time $T^* = T^1$. □

After adding a resource restriction to the previous example (we assume $B^W = 25{,}000$ and all $b_i = 6.25$ and $B_i^R = 2000$), we obtain the following result, in comparison with the case without the resource restriction. (See Table 7).

6. Concluding remarks

In this paper, we considered a more flexible policy compared with the existing JRP models for the one-warehouse, *n*-retailer system. We introduced the mathematical model and two efficient algorithms for the joint replenishment and delivery scheduling of the warehouse. These algorithms use optimality conditions of each decision variable and determine the best solution recursively. We also developed the hybrid GA and compared it with two recursive algorithms for 1600 randomly generated problems. Despite the fact that our GA is slightly inferior to the JRD–RAND, we showed the extension ability of our GA in dealing with resource restrictions- an important ability from the practical viewpoint. This policy can be extended to the joint inventory and routing problem for the one-warehouse, *n*-retailer system (Anily and Federgruen, 1993; Herer and Roundy, 1997). In this case, we expect that our GA can be modified and applied more easily to this problem compared with the two heuristic algorithms.

Acknowledgement

The authors are grateful to the constructive comments of the Editor-in-Chief and two anonymous referees. This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (The Regional Research Universities Program/Research Center for Logistics Information Technology).

References

Abdul-Jalbar, B., Gutierrez, J., Puerto, J., Sicilia, J., 2003. Policies for inventory/distribution systems: the effect of centralization vs. decentralization. Int. J. Prod. Eco. 81–82, 281–293.

Anily, S., Federgruen, A., 1993. Two-echelon distribution systems with vehicle routing costs and central inventories. Operat. Res. 41, 37–47.

Aytug, H., Khouja, M., Vergara, F., 2003. Use of genetic algorithms to solve production and operations management problems: a review. Int. J. Prod. Res. 41, 3955–4009.

Cha, B., Moon, I., 2004. Joint replenishment and delivery scheduling in a supply chain. IE Interf. 17, 90–96.

Chan, C., Cheung, B., Langevin, A., 2003. Solving the multi-buyer joint replenishment problem with a modified genetic algorithm. Transport. Res. Part B 37, 291–299.

Eum, S., Lee, Y., Jung, J., 2005. A producer's allocation policy considering buyers' demands in the supply chain. J. Kor. Inst. Ind. Eng. 31, 210–218.

Goyal, S., 1973. Determination of economic packaging frequency for items jointly replenished. Manage. Sci. 20, 232–238.

Goyal, S., 1974. Determination of optimum packaging frequency of items jointly replenished. Manage. Sci. 21, 436–443.

Goyal, S., 1975. Analysis of joint replenishment inventory systems with resource restriction. Operat. Res. Quart. 26, 197–203.

Goyal, S., Deshmukh, S., 1993. A note on the economic ordering quantity for jointly replenished items. Int. J. Prod. Res. 31, 2959–2961.

Graves, S., 1979. On the deterministic demand multi-product single-machine lot scheduling problem. Manage. Sci. 25, 276–280.

Herer, Y., Roundy, R., 1997. Heuristics for a one-warehouse multi-retailer distribution problem with performance bounds. Operat. Res. 45, 102–115.

Hong, S., Park, Y., 2006. A Comparison Study on Retailer-managed and Vendor-managed Inventory Policies in the Retail Supply Chain. J. Kor. Inst. Ind. Eng. 32, 383–392.

Kaspi, M., Rosenblatt, M., 1983. An improvement of Silver's algorithm for the joint replenishment problem. IIE Trans. 15, 264–269.

Kaspi, M., Rosenblatt, M., 1991. On the economic ordering quantity for jointly replenished items. Int. J. Prod. Res. 29, 107–114.

Khouja, M., Michalewicz, Z., Satoskar, S., 2000. A comparison between genetic algorithms and the RAND method for solving the joint replenishment problem. Prod. Plan. Control 11, 556–564.

Li, Q., 2004. Solving the multi-buyer joint replenishment problem with the RAND method. Comput. Ind. Eng. 46, 755–762.

Lu, L., Posner, M., 1994. Approximation procedures for the one-warehouse multi-retailer system. Manage. Sci. 40, 1305–1316.

Maxwell, W., Muckstadt, J., 1985. Establishing consistent and realistic reorder intervals in production–distribution systems. Operat. Res. 33, 1316–1341.

Moon, I., Cha, B., 2006. The joint replenishment problem with resource restrictions. Euro. J. Operat. Res. 173, 190–198.

Roundy, R., 1985. 98%-effective integer-ration lot-sizing for one-warehouse multi-retailer systems. Manage. Sci. 31, 1416–1430.

Roundy, R., 1986. 98%-effective lot-sizing rule for a multi-product, multi-stage production/inventory system. Math. Operat. Res. 11, 699–727.

Schwarz, L., 1973. A simple continuous review deterministic one-warehouse N-retailer inventory problem. Manage. Sci. 19, 555–566.

Shu, F., 1971. A simple method of determining order quantities in joint replenishments under deterministic demand. Manage. Sci. 17, 406–410.

Silver, E., 1975. Modifying the economic order quantity (EOQ) to handle coordinated replenishment of two or more items. Prod. Invent. Manage. 16, 26–38.

Silver, E., 1976. A simple method of determining order quantities in jointly replenishments under deterministic demand. Manage. Sci. 22, 1351–1361.

Silver, E., Pyke, D., Peterson, R., 1998. Inventory Management and Production Planning and Scheduling, 3rd ed. John Wiley and Sons, New York.

Van Eijs, M.J.G., 1993. A note on the joint replenishment problem under constant demand. J. Operat. Res. Soc. 44, 185–191.

Viswanathan, S., 1996. A new optimal algorithm for the joint replenishment problem. J. Operat. Res. Soc. 47, 936–944.

Viswanathan, S., 2002. On optimal algorithms for the joint replenishment problem. J. Operat. Res. Soc. 53, 1286–1290.

Viswanathan, S., Mathur, K., 1997. Integrating routing and inventory decisions in one-warehouse multi-retailer multi-product distribution systems. Manage. Sci. 43, 294–312.

Wildeman, R., Frenk, J., Dekker, R., 1997. An efficient optimal solution method for the joint replenishment problem. Euro. J. Operat. Res. 99, 433–444.