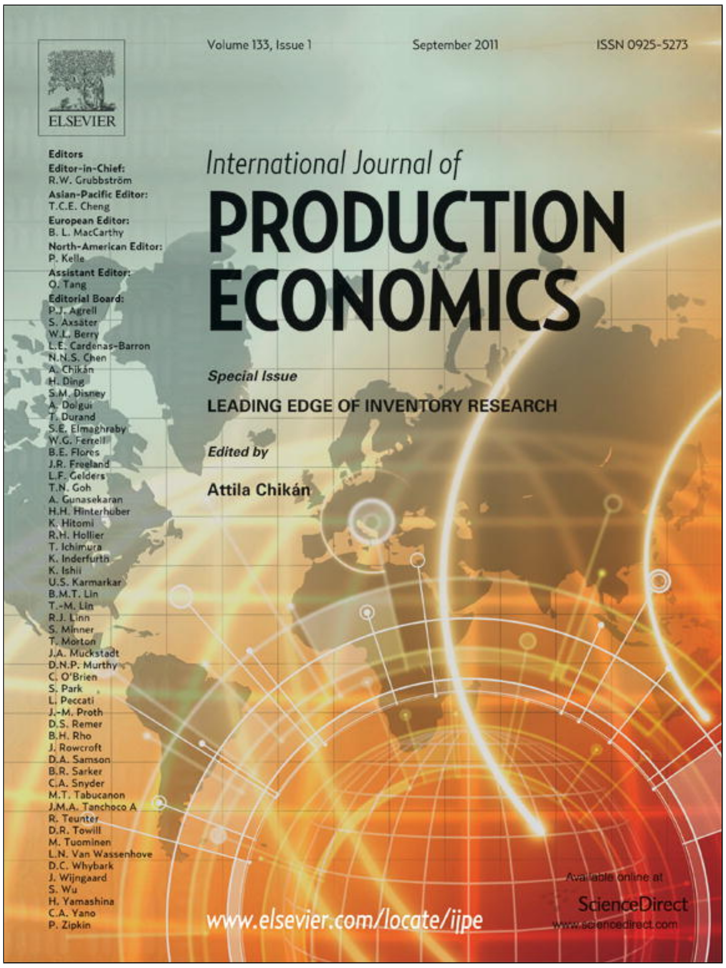
Volume 133, Issue 1
September 2011
ISSN 0925-5273
ELSEVIER
International Journal of
PRODUCTION ECONOMICS
Special Issue
LEADING EDGE OF INVENTORY RESEARCH
Edited by
Attila Chikán
www.elsevier.com/locate/ijpe
Available online at
ScienceDirect
www.sciencedirect.com

Int. J. Production Economics 133 (2011) 344–350

Contents lists available at ScienceDirect

Int. J. Production Economics

journal homepage: www.elsevier.com/locate/ijpe

The joint replenishment and freight consolidation of a warehouse in a supply chain

I.K. Moon [a,*], B.C. Cha [b], C.U. Lee [c]

[a] Department of Industrial Engineering, Pusan National University, Korea
[b] Department of Business Administration, Changwon National University, Korea
[c] Department of Industrial Systems & Information Engineering, Korea University, Korea

ARTICLE INFO

Article history:
Received 1 September 2008
Accepted 22 October 2009
Available online 2 February 2010

Keywords:
Joint replenishment problem
Freight consolidation
Heuristic algorithm

ABSTRACT

We have developed joint replenishment and consolidated freight delivery policies for a third party warehouse that handles multiple items, which have deterministic demand rates in a supply chain. Two policies are proposed and mathematical models are developed to obtain the optimal parameters for the proposed policies. Four efficient algorithms are presented to solve the mathematical models for the two policies. The performances of the two policies with the parameters obtained from the proposed algorithms are then compared with the common cycle approach for 1600 randomly generated problems. The results show the robust performance of the proposed algorithm for both policies.

1. Introduction

Supply Chain Management (SCM) is the management of materials 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 from the business community. In recent years, many companies have realized that significant cost savings can be achieved by integrating inventory control and delivery policies throughout their supply chains. Many operational policies and models have been developed in this area (Silver et al., 1998). The intent of these models is usually to minimize the total supply chain cost while satisfying stochastic customer demands.

In this paper, we focus on the joint replenishment and consolidated freight delivery scheduling of a warehouse in a supply chain. As shown in Fig. 1, a third party warehouse replenishes multiple items from multiple suppliers, and then delivers them to the customers who have ordered them in an E-marketplace. When a customer's order arrives via the E-marketplace, the warehouse must decide whether to ship the order immediately, or to wait for more orders to arrive so that the truck space (or container space depending on the transportation mode) is better utilized (thus, achieving an economy of scale). Furthermore, the recent introduction of a Vendor Managed Inventory (VMI) initiative (most noticeably in Walmart, as an example), particularly due to the availability of Electronic Data Exchange (EDI), enables coordination of supply chains by synchronizing inventory management and transportation planning (Cetinkaya and Lee 2000). The total supply chain cost for this type of warehouse can be significantly reduced by optimally determining: (1) the quantities and groupings of products to be replenished together; and (2) how often to deliver consolidated goods to customers.

The former decision problem (1) is known as the joint replenishment problem (JRP). After the early work of Shu (1971), the JRP has attracted much attention from researchers during the last three decades. Goyal (1974) proposed an enumeration approach to obtain an optimal solution to the JRP with constant unit costs. Goyal (1975) considered a single resource constraint and developed a heuristic algorithm using a Lagrangian multiplier. Silver (1975, 1976) discussed the advantages and the disadvantages of coordinating replenishments, and presented a very simple noniterative procedure for solving the JRP. Kaspi and Rosenblatt (1991) proposed a heuristic algorithm, where they first computed equally spaced values of the basic cycle time between a lower and an upper bound. Then, they applied the heuristic algorithm of Kaspi and Rosenblatt (1983), which is a modified version of Silver's (1975) algorithm for each value of the basic cycle time. They showed that their procedure (denoted as RAND) outperformed all the available heuristics. Van Eijs (1993) modified the lower bound on an optimal cycle time used to guarantee an optimal solution. Viswanathan (1996) suggested tighter bounds on the basic cycle time to improve the procedures by Goyal (1974). Wildeman et al. (1997) presented a new solution approach based on the Lipschitz optimization to obtain a solution with an arbitrarily small deviation from the optimum. Khouja et al. (2000) presented genetic algorithms for the JRP, and compared the performance of their genetic algorithms with that of the Kaspi and Rosenblatt's (1991) heuristics algorithm .Viswanathan (2002) proposed a modified

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

0925-5273/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.ijpe.2009.10.030

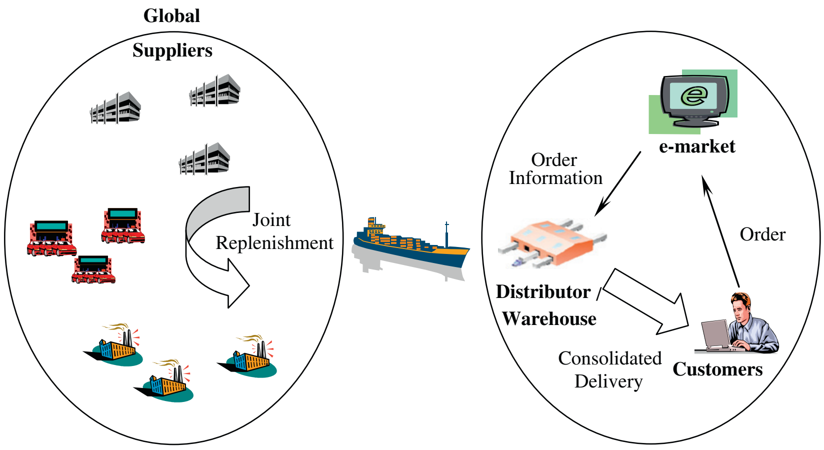

Fig. 1. Joint replenishment and delivery scheduling model in a supply chain.

algorithm to obtain the optimal strict cyclic policy. Recently, Li (2004) considered the multi-buyer joint replenishment problem and proposed a new, efficient RAND method. Moon and Cha (2006) developed both a modified RAND algorithm and a genetic algorithm for the joint replenishment problem with resource restrictions. Cha and Moon (2005) solved the joint replenishment problem with quantity discounts using both simple heuristics and a modified RAND algorithm denoted as QD-RAND. Cha et al. (2008) solved the joint replenishment and delivery scheduling of the one-warehouse, n-retailer system, and developed both heuristic algorithms and genetic algorithm.

Order consolidation reduces transportation costs due to the economies of scale inherent in freight distribution. Many firms have been consolidating cargos for a number of years as an effective way of reducing transportation costs. Consolidation has also become a subject of renewed interest as the difference between truckload and less-than-truckload rates has increased (Higginson and Bookbinder 1994). The consolidation policy determines how long and/or what quantities of shipments must be accumulated, by considering the trade-offs between the consolidation benefit of lower transportation charges and the consolidation penalties of increased inventory costs, longer customer waiting time, and increased terminal operating and ownership costs. Hall (1987) introduced three strategies for consolidation – inventory consolidation, vehicle consolidation and terminal consolidation – and found that consolidation can take place when vehicles make multiple pickups (collecting), at transshipment terminals. Gupta and Bagchi (1987) computed the minimum cost-effective lot size to be consolidated in a just-in-time procurement environment. Higginson and Bookbinder (1994) examined three release policies for shipment consolidation: (1) a time policy that ships at a prespecified time; (2) a quantity policy that ships when a given quantity is accumulated; and (3) a time/quantity policy that ships at a time earlier than the time and quantity values. Recently, Cetinkaya and Lee (2002) developed mathematical models for coordinating inventory and transportation decisions at a third party warehouse. However, they considered only a special case of this coordination problem, where the warehouse delivers a single item.

We will extend Cetinkaya and Lee's results to consider the joint replenishment of multiple items. Our objective is to develop the joint replenishment and consolidated freight delivery models for a warehouse. We consider n different types of items and assume that all the items are handled at the warehouse. We also assume that item i is stocked and delivered to the customers who order it. The warehouse replenishes item i at each integer multiple (k_i) of the basic cycle time (T). As in Cetinkaya and Lee's study, two different shipment release policies for freight consolidation are considered. The next section introduces a stationary policy (i.e., the interval between successive deliveries is a constant) and proposes two different heuristic algorithms to obtain the near optimal parameters for the policy by using the optimality condition for each decision variable. A quasi-stationary policy and two heuristic algorithms are presented in the third section, and the results of computational experiments are presented to characterize the performance of our proposed heuristics in the fourth section. Finally, we present our conclusions from the present work in the final section.

2. Stationary policy

The following notation is employed in this study.

i	item index, $i=1, 2, \ldots, n$
D_i	demand rate of item i (which is deterministic)
S^W	major ordering cost incurred at every basic cycle time
s_i^W	minor ordering cost incurred when item i is included in a group replenishment
h_i^W	inventory holding cost of item i per unit, per unit time
s_i^C	outbound transportation cost for item i
w_i^C	customer waiting cost for item i per unit, per unit time
T	basic cycle time for the warehouse (decision variable)
k_i	integer number that determines the replenishment schedule of item i (decision variable)
$\mathbf{K}$	$n\times 1$ vector that consists of k_i, $i=1, \ldots, n$
f_i	integer number that determines the outbound delivery schedule of item i (decision variable)
$\mathbf{F}$	$n\times 1$ vector that consists of f_i, $i=1, \ldots, n$

Under the stationary policy, the interval between successive deliveries of item i from a warehouse to the customer must remain the same throughout the planning horizon, as shown in Fig. 2 (where the consolidated freight order quantity also does not change). The warehouse replenishes item i at each integer multiple k_i of the basic cycle time T.

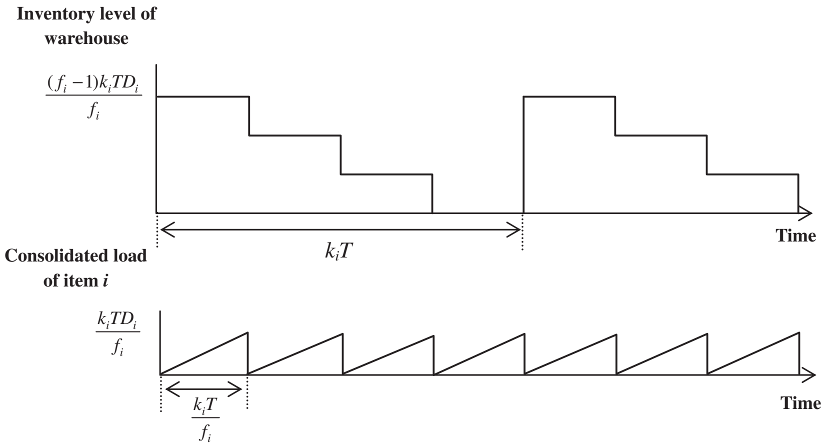

Fig. 2. Joint replenishment and consolidated delivery of item i under the stationary policy.

Using the above notation, the total relevant cost per unit time is given by

$$TC(T,K,F)=\frac{S^W+\sum_{i=1}^{n}\frac{s_i^W}{k_i}}{T}+\sum_{i=1}^{n}\frac{(f_i-1)k_iTD_ih_i^W}{2f_i}+\sum_{i=1}^{n}\frac{f_is_i^C}{k_iT}+\sum_{i=1}^{n}\frac{k_iTD_iw_i^C}{2f_i} \quad (1)$$

The total cost is the sum of the ordering cost (major and minor), the inventory holding cost, the outbound transportation cost, and the customer waiting cost. The values of T, $\mathbf{K}$, and $\mathbf{F}$ that minimize the total relevant cost per unit time follow the optimality conditions below.

For given values of $\mathbf{K}$ and $\mathbf{F}$, the total cost function is a convex function of T. Thus, the optimal basic cycle time T is obtained by taking the first order derivative of the total cost function, as given by Eq. (2).

$$T*=\left[\frac{2\left(S^W+\sum_{i=1}^{n}\frac{s_i^W+f_is_i^C}{k_i}\right)}{\sum_{i=1}^{n}k_iD_i\left(h_i^W+\frac{w_i^C-h_i^W}{f_i}\right)}\right]^{1/2} \quad (2)$$

For given values of f_i and T, the optimal value of k_i always satisfies the following:

$TC(k_i)\le TC(k_i+1)$ and $TC(k_i)\le TC(k_i-1)$

Using Eq. (1), an optimality condition of k_i is:

$$k_i(k_i-1)\le\frac{2(s_i^W+f_is_i^C)}{T^2D_i\left(h_i^W+\frac{w_i^C-h_i^W}{f_i}\right)}\le k_i(k_i+1) \quad (3)$$

For given values of k_i and T, the optimal value of f_i follows:

$TC(f_i)\le TC(f_i+1)$ and $TC(f_i)\le TC(f_i-1)$

Similarly, using Eq. (1), an optimality condition of f_i is:

$$f_i(f_i-1)\le\frac{k_i^2T^2D_i(w_i^C-h_i^W)}{2s_i^C}\le f_i(f_i+1) \quad (4)$$

If $h_i^W\ge w_i^C$, then f_i =1 as the total cost function is an increasing step function of f_i.

Using the above optimality conditions, we can develop a simple recursive algorithm to obtain the near optimal parameters for the stationary policy. The procedure of the heuristic algorithm is as follows:

Heuristic algorithm for the stationary policy (SP-H)

Step 1. Set the iteration number r=0. Set $k_i(r)$=1 and $f_i(r)$=1 for i=1, …, n.
Let $T(r)$=0. Go to Step 2.

Step 2. Set r=r+1 and go to Step 3.

Step 3. For given values of $\mathbf{K}(r-1)$ and $\mathbf{F}(r-1)$, compute the optimal T using Eq. (2).
Set $T(r)$=T. If $T(r)$=$T(r-1)$, stop. Otherwise, go to Step 4.

Step 4. For given values of $T(r)$ and $\mathbf{F}(r-1)$, compute the optimal k_i, i=1, …, n, using Eq. (3). Set $k_i(r)$=k_i, i=1, …, n, and go to Step 5.

Step 5. For given values of $T(r)$ and $\mathbf{K}(r)$, compute the optimal f_i, i=1, …, n, using Eq. (4). Set $f_i(r)$=f_i, i=1, …, n, and go to Step 2.

In general, in JRPs, iterative algorithms, such as SP-H, converge to local optimal solutions. To avoid this problem, Kaspi and Rosenblatt (1991) developed the RAND algorithm. They obtained several local optimal solutions from the iterative algorithm with different initial values of T, and selected the best solution among all the local optimal solutions. Their simulation studies show that the initial value of T has a strong influence on the quality of the obtained solution. Using this idea, we modify the RAND algorithm and develop a new algorithm for the stationary policy. The modified RAND algorithm for the stationary policy, denoted as SP-RAND, proceeds as follows:

Modified RAND algorithm for the stationary policy (SP-RAND)

Step 1. Compute the lower and the upper bounds for T:
$Tmax=[2(S+\sum_{i=1}^{n}s_i)/\sum_{i=1}^{n}D_ih_i]^{1/2}$ and
$T_{min}=min(2s_i/D_ih_i)^{1/2}$

Step 2. For given m, divide $[T_{min}, T_{max}]$ into m equally spaced values of T: $(T_1, \ldots, T_j, \ldots, T_m)$. Set j=0.

Step 3. Set j=j+1 and r=0. Let $T_j(r)$=T_j and $F(r)=(f_1(r),f_2(r),\ldots,f_n(r))=(1,1,\ldots,1)$.

Step 4. Set r=r+1.

Step 5. For given values of $T_j(r-1)$ and $\mathbf{F}(r-1)$, compute the optimal k_i, i=1, …, n, using Eq. (3). Set $k_i(r)$=k_i, i=1, …, n.

Step 6. For given values of $T_j(r-1)$ and $\mathbf{K}(r)$, compute the optimal f_i, i=1, …, n, using Eq. (4). Set $f_i(r)$=f_i, i=1, …, n.

Step 7. For given values of $\mathbf{K}(r)$ and $\mathbf{F}(r)$, compute the optimal T using Eq. (2).
Set $T_j(r)$=T.

Step 8. If $T_j(r)\ne T_j(r-1)$, go to Step 4.
Otherwise, set $T_j^*=T_j(r)$, $k_{ij}^*=k_i^*(r)$, and $f_{ij}^*=f_i^*(r)$.
Compute TC_j for this (T_j^*,K_j^*,F_j^*).

Step 9. If $j \neq m$, go to Step 3.
Otherwise, stop and select (T_j^*, K_j^*, F_j^*) with the minimum TC.

3. Quasi-stationary policy

The quasi-stationary policy is where the interval between successive deliveries of an item i is allowed to change over time, as shown in Fig. 3. Under this policy, we can take advantage of cross-docking; i.e., at the end of the replenishment cycle, the supplier ships R_iD_i units directly to the retailers. Let $\mathbf{R}$ denote an $n \times 1$ vector consisting of R_i, where $i=1, \ldots, n$; i.e., $\mathbf{R}=(R_1, \ldots, R_n)$.

Using the above definition and the notation provided in Section 2, the total relevant cost per unit time is given by

$$TC(T,K,F,R) = \frac{S^W + \sum_{i=1}^{n}(s_i^W/k_i)}{T} + \sum_{\{i:f_i \neq 1\}} \frac{f_i(k_iT-R_i)^2 D_i h_i^W}{2(f_i-1)k_iT} + \sum_{i=1}^{n}\frac{f_i s_i^C}{k_iT} + \sum_{\{i:f_i \neq 1\}} \frac{[(k_iT-R_i)^2+(f_i-1)R_i^2]D_i w_i^C}{2(f_i-1)k_iT} + \sum_{\{i:f_i=1\}} \frac{k_iTD_iw_i^C}{2} \tag{5}$$

where $\{i:f_i \neq 1\}$ is the set of items that satisfy $f_i \neq 1$.

The values of T, $\mathbf{K}$, $\mathbf{F}$, and $\mathbf{R}$ that minimize the total relevant cost per unit time follow the following optimality conditions.

For a given set of $\mathbf{K}$, $\mathbf{F}$, and $\mathbf{R}$, the total cost function is a convex function of T. Thus, the optimal basic cycle time T is easily obtained by taking the first order derivative of the cost function, as given by Eq. (6).

$$T^* = \left[\frac{S^W + \sum_{i=1}^{n}\frac{s_i^W+f_is_i^C}{k_i} + \sum_{\{i:f_i \neq 1\}}\frac{f_iR_i^2D_i(h_i^W+w_i^C)}{2(f_i-1)k_i}}{\sum_{\{i:f_i \neq 1\}}\frac{k_iD_i(f_ih_i^W+w_i^C)}{2(f_i-1)} + \sum_{\{i:f_i=1\}}\frac{k_iD_iw_i^C}{2}}\right]^{1/2} \tag{6}$$

For given values of f_i, R_i and T, the optimal value of k_i always follows:

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

If f_i is equal to one, the optimal k_i follows:

$$k_i(k_i-1) \leq \frac{2(s_i^W+f_is_i^C)}{T^2D_iw_i^C} \leq k_i(k_i+1) \tag{7}$$

Otherwise (if f_i is not equal to one), the optimal value of k_i follows:

$$k_i(k_i-1) \leq \frac{2(f_i-1)(s_i^W+f_is_i^C)+f_iR_i^2D_i(h_i^W+w_i^C)}{T^2D_i(f_ih_i^W+w_i^C)} \leq k_i(k_i+1) \tag{8}$$

For given values of k_i and T, the optimal values of f_i and R_i depend on each other, as in the following two cases:

Case 1: If f_i is equal to 1, then R_i equals k_iT .
Case 2: If f_i is greater than or equal to 2, then R_i is less than k_iT. In this case, for a given set of $\mathbf{K}$, $\mathbf{F}$, and T, the total cost function is a convex function of R_i. Thus, the optimal value of R_i is easily obtained by taking the first order derivative of the cost function.

The optimal value of R_i is calculated using Eq. (9).

$$R_i = \frac{(f_ih_i^W+w_i^C)k_iT}{f_i(h_i^W+w_i^C)} \tag{9}$$

For given values of k_i, R_i, and T, the optimal value of f_i follows:

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

Using Eq. (1), an optimality condition of f_i is:

$$(f_i-1)(f_i-2) \leq \frac{(k_iT-R_i)^2D_i(h_i^W+w_i^C)}{2s_i^C} \leq (f_i-1)f_i \tag{10}$$

For given values of k_i and T, the optimal values of f_i and R_i are obtained by selecting the values of f_i and R_i with the minimum total cost from the above two cases, i.e., we calculate

$$TC(f_i,R_i)-TC(f_i=1,R_i=k_iT) = \frac{(k_iT-R_i)^2(f_ih_i^W+w_i^C)D_i}{2(f_i-1)k_iT} + \frac{(f_i-1)s_i^C}{k_iT} + \frac{\{R_i^2-(k_iT)^2\}D_iw_i^C}{2k_iT} \tag{11}$$

If Eq. (11) is greater than zero, then $f_i=1$ and $R_i=k_iT$. Otherwise, the optimal values of f_i and R_i are calculated using Eqs. (9) and (10).

Using the above optimality conditions, we develop a simple recursive algorithm to obtain the optimal parameters for the quasi-stationary policy (QSP-H). The procedure of the heuristic algorithm is as follows:

Heuristic algorithm for the quasi-stationary policy (QSP-H)

Step 1. Set the iteration number r=0. Set $k_i(r)$=1 and $f_i(r)$=1 for $i=1, \ldots, n$.
Let $T(r)$=0, Go to Step 2.
Step 2. Set r=r+1 and go to Step 3.
Step 3. For given given values of $\mathbf{K}(r-1)$, $\mathbf{F}(r-1)$, and $\mathbf{R}(r-1)$, compute the optimal T using Eq. (6). Set $T(r)$=T. If $T(r)$=$T(r-1)$, stop. Otherwise, go to Step 4.
Step 4. For given given values of $T(r-1)$, $\mathbf{F}(r-1)$, and $\mathbf{R}(r-1)$, if $f_i(r-1)$ =1, compute the optimal k_i using Eq. (7). Otherwise, compute the optimal value of k_i using Eq. (8). Set $k_i(r)$=k_i and go to Step 5.

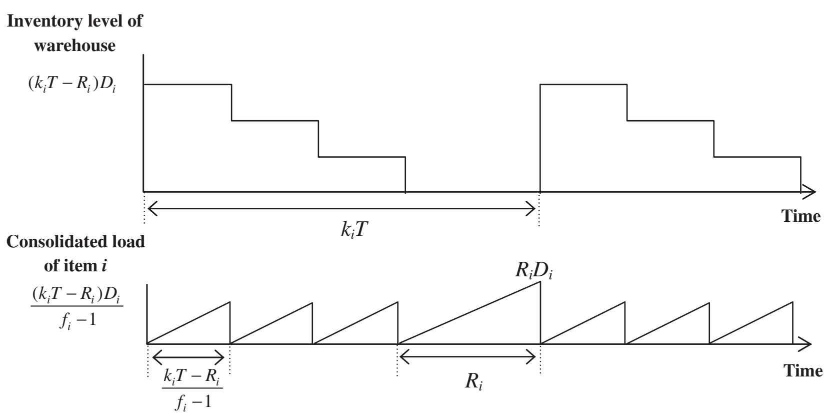

Fig. 3. Joint replenishment and consolidated delivery of item i under the quasi-stationary policy.

Step 5. For given given values of $T(r)$ and $\mathbf{K}(r)$, compute the optimal values of f_i and R_i, i=1, …, n, by the following sub-routine. Go to Step 2.

[sub-routine for item *i*]

Step 5-1. f_i =2
Step 5-2. old_ f_i=f_i
Step 5-3. For given values of $T_j(r)$, $k_i(r)$, and f_i, compute R_i using equation (9)
Step 5-4. For given values of $T_j(r)$, $k_i(r)$, and R_i, compute f_i using equation (10)
If old_ $f_i \neq f_i$, go to Step 5-2.
Otherwise, compute Eq. (11).
If Eq. (11) < 0, set $f_i(r)$=f_i and $R_i(r)$=R_i.
Otherwise, set $f_i(r)$=1 and $R_i(r)$=$k_i(r)T_j(r)$.
Exit.

Similar to the SP-RAND procedure described in Section 2, motivated by the RAND algorithm, we develop a new improved algorithm for the quasi-stationary policy. The modified RAND algorithm for the quasi-stationary policy, which we denote as QSP-RAND, proceeds as follows:

Modified RAND algorithm for the quasi-stationary policy (QSP-RAND)

Step 1. Compute the lower and the upper bounds for T:
$$Tmax = [2(S+\sum_{i=1}^{n} s_i)/\sum_{i=1}^{n} D_i h_i]^{1/2}$$
and $T_{min} = min(2s_i/D_i h_i)^{1/2}$

Step 2. For given m, divide [T_{min}, T_{max}] into m equally spaced values of T: (T_1, ⋯⋯⋯, T_j, ⋯⋯⋯, T_m). Set j=0.

Step 3. Set j=j+1 and r=0.
Set $T_j(r)$=T_j, $F(r)=(f_1(r),f_2(r),\ldots,f_n(r))=(1,1,\ldots,1)$ and
$R(r)=(R_1(r),R_2(r),\ldots,R_n(r))=(T_1(r),T_2(r),\ldots,T_n(r))$.

Step 4. Set r=r+1.

Step 5. For given values of $T_j(r-1)$, $\mathbf{F}(r-1)$, and $\mathbf{R}(r-1)$, if $f_i(r-1)$ =1, compute the optimal k_i using Eq. (7). Otherwise, compute the optimal k_i using equation (8).
Set $k_i(r)$=k_i, i=1, …, n.

Step 6. For given values of $T_j(r-1)$ and $\mathbf{K}(r)$, compute the optimal f_i and R_i, i=1, …, n, by the following sub-routine.

[sub-routine for item *i*]

Step 6-1. f_i=2
Step 6-2. old_ f_i=f_i
Step 6-3. For given given values of $T_j(r-1)$, $k_i(r)$, and f_i, compute R_i using Eq. (9)
Step 6-4. For given given values of $T_j(r-1)$, $k_i(r)$ and R_i, compute f_i using Eq. (10)
If old_ $f_i \neq f_i$, go to Step 6-2.
Otherwise, compute Eq. (11).
If Eq. (11) < 0, set $f_i(r)$=f_i and $R_i(r)$=R_i.
Otherwise, set $f_i(r)$=1 and $R_i(r)$=$k_i(r)T_j(r-1)$.
Exit.

Step 7. For given given values of $\mathbf{K}(r)$, $\mathbf{F}(r)$ and $\mathbf{R}(r)$, compute the optimal T using equation (6). Set $T_j(r)$=T.

Step 8. If $T_j(r) \neq T_j(r-1)$, go to Step 4.
Otherwise, set$T_j^* = T_j(r)$, $k_{ij}^* = k_i^*(r)$, $f_{ij}^* = f_i^*(r)$, and $R_{ij}^* = R_i^*(r)$.
Compute TC_j for this $(T_j^*, K_j^*, F_j^*, R_j^*)$.

Step 9. If $j \neq m$, go to Step 3.
Otherwise, stop and select $(T_j^*, K_j^*, F_j^*, R_j^*)$ with the minimum TC.

A numerical example will be employed to compare the proposed heuristic algorithms. The parameter values for this example are provided in Table 1. We also assume that S^W=$200.

The solutions and the total cost obtained using the four proposed algorithms and the two common cycle approaches (SP-CC and QSP-CC) are compared in Table 2, which shows that the quasi-stationary policy provides significantly lower total costs than the stationary policy, and that our proposed algorithms consistently provide better solutions than the common cycle approaches.

4. Computational experiments

In this section, we compare the solutions obtained using the four algorithms for 1600 randomly generated problems. The

Table 1
The parameter values for our example.

Item i	1	2	3	4	5	6
D_i	10,000	5,000	3,000	1,000	600	200
s_i^W	45	46	47	44	45	47
h_i^W	1	1	1	1	1	1
s_i^C	5	5	5	5	5	5
w_i^C	1.5	1.5	1.5	1.5	1.5	1.5

Table 3
Range of the parameter values.

D_i	s_i^W	s_i^C	h_i^W	w_i^C
U[500, 5000]	U[30, 50]	U[0.1s_i^W, 0.3s_i^W]	U[0.5, 3.0]	U[1.2h_i^W, 2.0h_i^W]

Table 2
A comparison of the solutions and the total costs for our numerical example.

Algorithm	T	k_i	f_i	R_i	TC	%
SP-CC	0.2215	1, 1, 1, 1, 1, 1	5, 4, 3, 2, 1, 1	–	$5001.31	21.51
SP-H	0.1973	1, 1, 1, 1, 2, 3	4, 3, 2, 1, 2, 2	–	$4850.39	17.85
SP-RAND	0.1881	1, 1, 1, 2, 2, 4	4, 3, 2, 3, 2, 2	–	$4828.89	17.33
QSP-CC	0.2772	1, 1, 1, 1, 1, 1	8, 6, 4, 2, 2, 1	0.1317, 0.1386, 0.1525 0.1940, 0.1940, 0.2772	$4249.56	3.25
QSP-H	0.2568	1, 1, 1, 1, 2, 3	8, 5, 4, 2, 4, 3	0.1220, 0.1335, 0.1412 0.1798, 0.2825, 0.4622	$4129.18	0.32
QSP-RAND	0.2414	1, 1, 1, 2, 2, 4	7, 5, 4, 4, 3, 4	0.1172, 0.1255, 0.1328 0.2655, 0.2897, 0.5310	$4115.81	–

Table 4
The values of the error (as a percentage) above the minimum total cost using the SP-RAND procedure (m=4n).

n	S^W	SP-CC			SP-H			SP-RAND m=0.5			SP-RAND m=n			SP-RAND m=2n		
		Num	Max	Avg.	Num	Max	Avg.	Num	Max	Avg.	Num	Max	Avg.	Num	Max	Avg.
10	100	9	4.498	1.361	56	0.843	0.086	96	0.155	0.002	97	0.155	0.002	100	0.000	0.000
	200	16	2.709	0.745	68	0.668	0.034	97	0.121	0.001	98	0.121	0.001	100	0.000	0.000
	300	30	1.662	0.453	73	0.332	0.032	98	0.152	0.002	99	0.152	0.002	100	0.000	0.000
	400	35	1.321	0.285	84	0.251	0.021	98	0.009	0.000	99	0.009	0.000	100	0.000	0.000
20	100	0	4.506	2.116	21	0.954	0.225	90	0.093	0.003	95	0.041	0.001	97	0.041	0.001
	200	0	3.542	1.438	39	0.749	0.092	85	0.143	0.007	96	0.040	0.001	96	0.040	0.001
	300	1	2.845	1.035	43	0.497	0.058	93	0.125	0.003	99	0.041	0.000	99	0.041	0.000
	400	2	2.260	0.766	50	0.308	0.037	96	0.084	0.001	97	0.007	0.000	99	0.005	0.000
30	100	0	4.480	2.453	8	1.126	0.250	77	0.176	0.008	92	0.076	0.003	98	0.010	0.000
	200	0	3.403	1.864	15	0.804	0.131	81	0.084	0.006	92	0.066	0.002	96	0.032	0.001
	300	0	2.802	1.466	30	0.484	0.082	88	0.037	0.002	96	0.022	0.001	100	0.000	0.000
	400	1	2.362	1.172	37	0.455	0.055	93	0.048	0.001	99	0.011	0.000	99	0.011	0.000
50	100	0	4.765	2.644	2	1.133	0.365	73	0.069	0.006	91	0.069	0.002	95	0.044	0.001
	200	0	4.096	2.181	4	0.717	0.229	83	0.061	0.003	90	0.051	0.001	98	0.019	0.000
	300	0	3.648	1.828	5	0.493	0.146	84	0.076	0.004	97	0.041	0.001	99	0.041	0.000
	400	0	3.226	1.557	8	0.369	0.098	78	0.065	0.003	92	0.022	0.001	98	0.006	0.000
Max.		35	4.765		84	1.133		98	0.176		99	0.155		100	0.044	
Avg.		6		1.460	34		0.121	88		0.003	96		0.001	98		0.000

Table 5
The error (as a percentage) above the minimum cost using the QSP-RAND procedure (m=4n).

N	SW	QSP-CC			QSP-H			QSP-RAND m=0.5			QSP-RAND m=n			QSP-RAND m=2n		
		Num	Max	Avg.	Num	Max	Avg.	Num	Max	Avg.	Num	Max	Avg.	Num	Max	Avg.
10	100	2	4.576	1.655	32	1.878	0.414	79	0.961	0.052	90	0.460	0.016	98	0.103	0.002
	200	12	2.660	0.880	45	1.846	0.232	87	0.716	0.028	94	0.351	0.011	99	0.281	0.003
	300	16	1.854	0.580	56	0.921	0.114	91	0.650	0.020	97	0.149	0.003	100	0.000	0.000
	400	25	1.854	0.401	64	0.641	0.092	90	0.336	0.015	95	0.336	0.007	100	0.000	0.000
20	100	0	5.217	2.261	7	2.850	0.859	69	0.770	0.060	86	0.375	0.019	97	0.350	0.004
	200	0	3.863	1.594	17	1.786	0.480	77	0.328	0.028	90	0.290	0.012	99	0.024	0.000
	300	2	2.557	1.157	18	1.610	0.314	79	0.385	0.021	90	0.237	0.008	98	0.124	0.001
	400	1	2.405	0.911	27	1.334	0.170	89	0.214	0.008	98	0.052	0.001	99	0.052	0.001
30	100	0	4.664	2.274	4	2.210	0.757	64	0.390	0.041	84	0.285	0.017	96	0.071	0.001
	200	0	3.388	1.840	5	1.630	0.524	74	0.237	0.027	88	0.193	0.008	96	0.185	0.004
	300	0	2.828	1.517	8	1.277	0.417	73	0.212	0.025	89	0.212	0.009	98	0.055	0.001
	400	0	2.638	1.307	7	1.175	0.305	68	0.371	0.018	79	0.210	0.011	90	0.113	0.005
50	100	0	4.932	2.470	1	2.654	0.895	53	0.299	0.038	82	0.267	0.013	92	0.125	0.004
	200	0	4.648	2.127	2	2.108	0.799	64	0.189	0.024	85	0.138	0.007	96	0.053	0.001
	300	0	4.198	1.827	3	1.991	0.636	66	0.173	0.021	82	0.141	0.008	95	0.060	0.001
	400	0	3.235	1.617	4	1.571	0.449	67	0.219	0.022	86	0.115	0.009	94	0.089	0.002
Max.		25	5.217		64	2.850		91	0.961		98	0.460		100	0.350	
Avg.		4		1.526	19		0.466	74		0.028	88		0.010	97		0.002

parameter values are all generated from a uniform distribution, as shown in Table 3.

Four different values for the number of items, n=10, 20, 30, and 50, and four different values for the major set-up cost, S^W=100, 200, 300, and 400 are considered. This results in 16 combinations of n and S^W, and for each combination, 100 problems with random parameter values (see Table 3) are generated and solved using the six algorithms (SP-CC, SP-H, SP-RAND, QSP-CC, QSP-H, and QSP-RAND). To compare the quality of the solutions using both the SP-RAND and QSP-RAND procedures with different values of m, four different values of m (m=0.5n, n, 2n, and 4n) are employed. The computational results are shown in Tables 4–6.

Table 4 shows how much (as a percentage) the total cost obtained using each algorithm exceeds the minimum total cost using the SP-RAND procedure (m=4n). The performance of the SP-RAND procedure improves as the value of m increases. This is consistent with the experimental results of Kaspi and Rosenblatt (1991). For n=20, 30, and 50, a value of m=n is sufficient as the maximum error is only 0.076%. For n=10, a higher value of m ($m \geq 4n$) can be considered, as the computational time is negligible. It also shows that our algorithms are significantly more efficient than the common cycle approach for the stationary policy (the maximum error was 4.765% and the average error was 1.460%).

Table 5 shows how much (as a percentage) the total cost obtained using each algorithm exceeds the minimum total cost obtained using the QSP-RAND procedure (m=4n). The performance of the QSP-RAND algorithm also improves as the value of m increases. For n=20, 30, and 50, a higher value of m has to be considered for the QSP-RAND procedure than for the SP-RAND procedure. Although the maximum error was 0.350% for m=2n,

Table 6.
The Error (as a percentage) above the minimum cost using the QSP-RAND procedure (m=4n).

n	S^W	SP-CC		SP-RAND		QSP-CC	
		Max	Avg.	Max	Avg.	Max	Avg.
10	100	17.362	12.078	12.384	10.573	4.576	1.655
	200	17.099	12.775	14.010	11.941	2.660	0.880
	300	16.761	13.458	14.852	12.947	1.854	0.580
	400	17.010	14.032	15.919	13.708	1.854	0.401
20	100	15.810	12.085	11.600	9.761	5.217	2.261
	200	15.368	12.227	12.718	10.635	3.863	1.594
	300	15.455	12.518	13.171	11.364	2.557	1.157
	400	15.517	12.798	13.563	11.940	2.405	0.911
30	100	14.534	11.982	10.755	9.301	4.664	2.274
	200	14.323	12.149	11.518	10.097	3.388	1.840
	300	14.247	12.319	11.966	10.696	2.828	1.517
	400	14.098	12.520	12.157	11.216	2.638	1.307
50	100	15.011	11.921	10.527	9.038	4.932	2.470
	200	14.876	11.928	10.673	9.539	4.648	2.127
	300	14.620	11.971	11.016	9.960	4.198	1.827
	400	14.233	12.039	11.366	10.320	3.235	1.617
Max.		17.362		15.919		5.217	
Avg.			12.425		10.815		1.526

the average error was only 0.002%. Therefore, for higher values of n, a value of m=2n is adequate. For smaller values of n, a value of $m \geq 4n$ can also be employed, as the computational time is negligible. It also shows that the proposed algorithm provides significantly better solutions than the common cycle approach for the quasi-stationary policy (the maximum error is 5.217% and the average error is 1.526%).

Table 6 shows a comparison of the total cost obtained using the four algorithms: SP-CC, SP-RAND, QSP-CC, and QSP-RAND. The QSP-RAND algorithm outperforms all the other algorithms. We can find that the quasi-stationary policy reduces the total cost significantly when compared to the stationary policy (the maximum reduction in cost is 15.919% lower, and on average 10.815% lower). This implies that a relaxation of fixed delivery intervals at the end of the replenishment cycle under the quasi-stationary policy significantly reduces transportation costs without significantly affecting customer service (delivery commitment).

5. Concluding remarks

We have developed joint replenishment and consolidated freight delivery models for a warehouse that handles multiple products with emphasis on customer service (e.g., the E-marketplace). We introduced two time-based policies for the warehouse and developed four efficient algorithms to obtain the near optimal parameters for these policies. Our algorithms are based on the optimality conditions of the decision variables. Using comprehensive computational experiments, we have shown that the performance of the SP-RAND and QSP-RAND algorithms improved as the value of m increased. For both the SP-RAND and QSP-RAND algorithms, proper values of m are proposed with respect to the number of items. We also showed that the quasi-stationary policy incurs significantly lower total costs than the stationary policy by achieving further scales of transportation economy without significantly affecting customer service. For both policies, the proposed algorithms consistently provided better solutions than the common cycle approach. These policies can easily be implemented to a single-warehouse, multi-retailer problem to streamline supply chain operations.

.

Acknowledgments

The authors are grateful to the constructive comments of two anonymous referees. This work was supported by the Grant of the Korean Ministry of Education, Science and Technology (The Regional Core Research Program/Institute of Logistics Information Technology).

References

Cetinkaya, S., Lee, C., 2002. Optimal outbound dispatch policies: modeling inventory and cargo capacity. Naval Research Logistics 49, 531–556.
Cha, B., Moon, I., 2005. The joint replenishment problem with quantity discounts. OR Spectrum 27, 569–581.
Cha, B., Moon, I., Park, J., 2008. The joint replenishment and delivery scheduling of the one-warehouse n-retailer system. Transportation Research-Part E 44, 720–730.
Goyal, S., 1974. Determination of optimum packaging frequency of items jointly replenished. Management Science 21, 436–443.
Goyal, S., 1975. Analysis of joint replenishment inventory systems with resource restriction. Operations Research Quarterly 26, 197–203.
Gupta, Y., Bagchi, P., 1987. Inbound freight consolidation under just-in-time procurement: application of clearing models. Journal of Business Logistics 8, 74–94.
Hall, R., 1987. Consolidation strategy: inventory, vehicles and terminals. Journal of Business Logistics 8, 57–73.
Higginson, J., Bookbinder, J., 1994. Policy recommendations for a shipment-consolidation program. Journal of Business Logistics 15, 87–112.
Kaspi, M., Rosenblatt, M., 1983. An improvement of Silver's algorithm for the joint replenishment problem. IIE Transactions 15, 264–269.
Kaspi, M., Rosenblatt, M., 1991. On the economic ordering quantity for jointly replenished items. International Journal of Production Research 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. Production Planning & Control 11, 556–564.
Li, Q., 2004. Solving the multi-buyer joint replenishment problem with the RAND method. Computers & Industrial Engineering 46, 755–762.
Moon, I., Cha, B., 2006. The joint replenishment problem with resource restrictions. European Journal of Operational Research 173, 190–198.
Shu, F., 1971. A simple method of determining order quantities in joint replenishments under deterministic demand. Management Science 17, 406–410.
Silver, E., 1975. Modifying the economic order quantity (EOQ) to handle coordinated replenishment of two or more items. Production & Inventory Management 16, 26–38.
Silver, E., 1976. A simple method of determining order quantities in jointly replenishments under deterministic demand. Management Science 22, 1351–1361.
Silver, E., Pyke, D., Peterson, R., 1998. Inventory Management and Production Planning and Scheduling, 3rd ed. John Wiley & Sons, New York.
Van Eijs, M., 1993. A note on the joint replenishment problem under constant demand. Journal of the Operational Research Society 44, 185–191.
Viswanathan, S., 1996. A new optimal algorithm for the joint replenishment problem. Journal of the Operational Research Society 47, 936–944.
Viswanathan, S., 2002. On optimal algorithms for the joint replenishment problem. Journal of the Operational Research Society 53, 1286–1290.
Wildeman, R., Frenk, J., Dekker, R., 1997. An efficient optimal solution method for the joint replenishment problem. European Journal of Operational Research 99, 433–444.