東海學報 42 卷(2001):23-39
中華民國九十年七月出版

On the Joint Replenishment Problem under General-Integer Policy

Ming-Jong Yao (姚銘忠) and Fang-Chuan Lee (李芳娟)

Department of Industrial Engineering and Enterprise Information
Tung-Hai University

ABSTRACT

This study performs theoretical analysis on the Joint Replenishment Problem (*JRP*) under General-Integer (*GI*) policy. The *JRP* models concern how to determine lot sizes and to schedule replenishment times for products so as to minimize the total costs per unit time. *GI* policy requires replenishment frequency of each product, denoted by k_i, to be a general integer, *i.e.*, $k_i = 1, 2, 3, \ldots$. In this study, we utilize a 10-product example to graphically present the curve of the optimal total cost with respect to the values of basic period. Under *GI* policy, we discover an interesting property on the optimal curve for the *JRP,* and we prove that the optimality structure of the *JRP* is *piece-wise convex.* By making use of the junction points in the optimality structure, we derive an effective (polynomial-time) search algorithm to secure a global solution for the *JRP* under *GI* policy. Evidently, we provide a numerical example to demonstrate the efficiency of the proposed algorithm.

***Key words*: Inventory, Scheduling, Lot size, Global optimum**

1. Introduction

1.1 Background and problem description

The Joint Replenishment Problem (*JRP*) is concerned with the determination of lot sizes and schedule of *n* products in single-facility production/inventory systems over an infinite (and continuous) planning horizon.

The objective of the *JRP* is to minimize the total costs incurred per unit time. The costs considered generally include setup costs and inventory holding costs. For each product *i*, its annual demand d_i is fixed (and continuous), and each unit incurs of holding cost h_i each year. The *JRP* assumes that the single facility has infinite capacity, and therefore, the replenishment for each product is instantaneous. Also, in the *JRP*, a product must be packaged immediately after it is manufactured by the production facility. Therefore, no holding cost incurs for the raw material. On the other hand, two types of setup costs are considered in the *JRP*:

1. A *major setup cost*, denoted by *A*, incurs whenever the production facility sets up to jointly replenish a subset of products.
2. A *minor setup cost* a_i is incurred while each product *i* is replenished (manufactured and packaged).

We note that major and minor setup costs are usually *independent of* the quantities of the products jointly replenished.

For decision makers facing the *JRP*, an intuitive move is to jointly replenish many products in each major setup to share the major setup cost so as to minimize the average total costs. Therefore, the focus of the *JRP* is to coordinate the replenishment schedule of each product *i* to economically share major setup cost, and balance the holding costs from the inventory of jointly replenished products.

In general, companies invest about 30% of their current asset and 90% of their operational capital on the inventories (*i.e.*, raw materials, purchased parts and work-in-process, etc.; see Stevenson 1993). Major setups often incur significant setup times and costs in certain industries; for instance, pharmaceutical, chemical processes, and textile companies. If the executive managers may effectively apply the concept of the *JRP* in their production/inventory systems, they could joint the replenishment schedule of the products to satisfactorily meet customers' demand and importantly, reduce significant cost in the meanwhile.

The solution approaches for the *JRP* usually assume that each product *i* is replenished after a fixed cycle, denoted by T_i where T_i is the length of time between two consecutive minor setups for product *i*.

Most of early studies assume that T_i is equal to a positive integer k_i times *B*, *i.e.*, $T_i = k_i B$, where *B* is a basic period in the production planning horizon. Also, it is usually assumed that the replenishment frequency for major setup, denoted by k_0, is always set to 1 in the *JRP*. In order to secure the optimal solution, one must simultaneously determine the value of *B* and the set of optimal (integer) multipliers $\{k_i: i = 1, 2, \ldots, n\}$ in the *JRP*. One may refer to Goyal and Satir (1989) as well as Aksoy and Erenguc (1988) for the details on the problem definition and the early studies of the *JRP*.

Based on the assumptions discussed above, the mathematical model for the *JRP* is formulated as follows.

$$\textit{Minimize } TC_{GI}(\{k_i\}, B) = \frac{1}{B}\left(\frac{A}{k_0} + \sum_{i=1}^{n} \frac{a_i}{k_i}\right) + \frac{B}{2}\sum_{i=1}^{n} k_i d_i h_i \qquad (1a)$$

$$\text{subject to} \quad k_i \in \{1, 2, 3, \cdots\}, k_0 = 1. \qquad (1b)$$

The subscript *GI* in the objective function $TC_{GI}(\{k_i\},B)$ indicates that the *JRP* model is formulated under *General Integer* (*GI*) *policy*, which is expressed by constraint (*1b*). *GI* policy requires that all the k_i's must be positive integers. Therefore, the *JRP* model in (1) is a nonlinear, integer problem.

1.2 Literature survey

Arkin, Joneja, and Roundy (1989) proved that the *JRP* is a *NP-hard* problem, *i.e.*, the *JRP* is not solvable by polynomial-time algorithms. The *JRP* has been studied for some thirty years. Extensive research efforts have been addressed to attempt efficient heuristics for solving the *JRP*.

Early, Shu (1971), Nocturne (1973) and Goyal (1973a) pioneered the research for the *JRP*. They solved the *JRP* by simply dividing the products into only *two* groups. Shu (1971) and Nocturne (1973) solved the *JRP* using graphical heuristics. Goyal (1973a) introduced another simple heuristic that tries to secure B^*, the optimal value of *B*, by the first derivative of $TC_{GI}(\{k_i\},B)$ and used simple rule to decide k_1^* and k_2^* where k_1^* and k_2^* are the optimal replenishment frequencies for those two groups.

Goyal (1973b) initiated the research to derive heuristics for the *JRP* that divide *n* products into *more than* two groups. Goyal (1974a, b) proposed an enumeration approach, and he claimed that it always secure a global optimal solution (though without proof). The essence of his enumeration approach is as follows. Enumerate *B* and k_i for each product *i* so as to satisfy both of the following conditions:

$$TC_{GI}(k_i(B),B) \le TC_{GI}(k_i(B)+1,B) \qquad (2a)$$

$TC_{GI}\,(k_i(B),B) < TC_{GI}\,(k_i(B)\text{-}1,B)$. (2*b*)

Later, Goyal (1988) and van Eijs (1993) presented examples that showed that the conditions in (4) are not sufficient conditions for securing a global optimal solution for the *JRP*. Meanwhile, van Eijs (1993) derived another algorithm that improves efficiency of Goyal's (1974b) algorithm. The shortcoming for this category of solution approaches is that one needs tremendous search efforts to enumerate between the upper and lower bounds on *B*. The run time of these approaches grows exponentially with the number of products. If there are more than 10 products in the *JRP*, extremely long run time make these enumeration approaches impractical. To address to this concern, Viswanathan (1996) derived some theoretical results that shorten the range between the upper and lower bounds on *B*. Viswanathan also presented an efficient algorithm that usually secures a "reasonable good" solution with a very shorter run time.

Another category of the solution approaches for the *JRP* are non-iterative procedures. First, Silver (1976) derived some analytical properties for the *JRP* under the assumption that all the $k_i^{'}s$ are continuous variables. Then, he solved the optimal multipliers k_i^*'s in closed-form. The three steps in Silver's procedure are: (1) Set as product number 1 for the product with the minimal value of $\frac{a_i}{h_i d_i}$. (2) Secure k_i^* by (a) computing a value of k_i by $k_i = \sqrt{\frac{a_i}{d_i h_i}}\sqrt{\frac{d_1 h_1}{A+a_1}}$, and (b) rounding the value of k_i to the nearest positive integer. (3) Secure the optimal value of *B* using the $\{k_i^*\}$ secured from the second step. Goyal and Belton (1979) suggested to improve Silver's procedure by changing the criterion value (of picking product 1) from $\frac{a_i}{d_i h_i}$ to $\frac{A+a_i}{d_i h_i}$. They provided an example that shows their modified procedure secures a better solution than Silver's procedure. Later, Kaspi and Rosenblatt (1983) brought another example to demonstrate that Goyal and Belton's procedure secures a poorer solution than Silver's. In order to improve Goyal and Belton's solution, Kaspi and Rosenblatt (1983) suggested to recalculated $\{k_i\}$ and *B* after Goyal and Belton's procedure secures its solution. Based on similar philosophy, Jackson, Maxwell and Muckstadt (1985) derived some interesting theorems on the optimal grouping for the products using $\frac{a_i}{d_i h_i}$ as the criterion value.

1.3 The motivation to study the *JRP* under *PoT* policy

In literature, few research efforts have been addressed to explore the optimal structure of the *JRP* in literature. And, to the best of the authors' knowledge, no solution approach is able to guarantee to secure an optimal solution for the *JRP*.

Before our study, we found only two papers that tried to explore the optimal structure of the *JRP*. In Viswanathan's (1996) paper, he implies that the optimal objective value of the *JRP* is piece-wise convex on *B*, though he did not bring the proof for this assertion. Also, since his theoretical results provide insufficient information to reveal the overview for the optimal structure of the *JRP* (under *GI* policy), Viswanathan's solution approach secures only an approximate solution (that is close to the real optimal solution.) Besides, Wildeman et al. (1997) did some decent studies on a relaxed version of the *JRP* in which they replaced $k_i \in \{1, 2, \ldots\}$ in (*1b*) with $k_i \geq 1$ and $k_i \in \Re$ for all i =1, 2,… , *n*. By utilizing their theoretical results, they derived tight upper and lower bounds on the search range. They proposed to use a *dynamic Lipschitz optimization procedure* to secure an approximate solution. But, neither Viswanathan (1996) nor Wildeman et al. (1997) guarantees that their algorithm is able to secure a global optimal solution.

To fulfill the research gap discussed above, we would like to investigate the optimality structure of the *JRP*. Also, based on the optimality structure of the *JRP*, we derive an effective search algorithm that is able to secure a global optimal solution in this paper.

The rest of this paper is organized as follows. In Section 2, we will present a full theoretical analysis on the optimal cost function for the *JRP*. Based on the theoretical results in Section 2, we derive an effective search algorithm to secure a global solution for the *JRP* under *GI* policy in Section 3. Evidently, we provide a numerical example to demonstrate the efficiency of the proposed algorithm in Section 4. Finally, we address our concluding remarks in Section 5.

2. Theoretical Analysis on the Optimal Cost Function

In this section, we first discuss some remarks and propositions to provide insights into the TC_{GI} function. Next, we introduce the “junction points” in the curve of the TC_{GI} function and demonstrate how to efficiently locate the junction points of the TC_{GI} function. These junction points assist us in securing the set of optimal multipliers for each given value of B that facilitates the derivation of the search algorithm presented in Section 3.

Under *GI* policy, the cost expression for a product i, is,

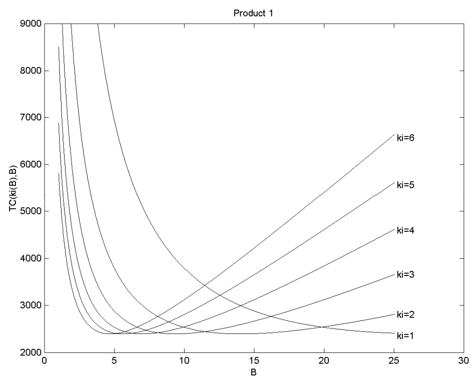
$$TC_i\,(k_i, B) = \frac{a_i}{k_i B} + \frac{h_i}{2} d_i k_i B \qquad (3)$$

where $k_i \geq 1$; k_i : integer, $i = 1, 2, \ldots, n$. For a given B, one may secure the optimal multiplier k_i so as to $TC_i(k_i,B)$. We denote it as $\underline{TC_i}(B)$, the minimum cost function with respect to B for product i, *i.e.*,

$$\underline{TC}_i\,(B) = \min_{k_i} \{TC_i\,(k_i, B)\}. \qquad (4)$$

2.1 Some insights into the optimal cost function

Denote TC_{GI} (B) as the optimal cost function of the *JRP* at a given B, *i.e.*, $TC_{GI}(B) = \frac{A}{k_0 B} + \sum_{i=1}^{n} \underline{TC}_{GI,i}(B)$ where $k_0 = 1$. We use the data in Section 4 to plot the optimal cost curve of the $\underline{TC_i}(B)$ function with respect to B for products 1 and 2 in Figure 1. Importantly, Figure 1 shows an interesting property on the $\underline{TC}_i$ (B) function as follows. The following theoretical results provide us some insights into the TC_{GI} function.



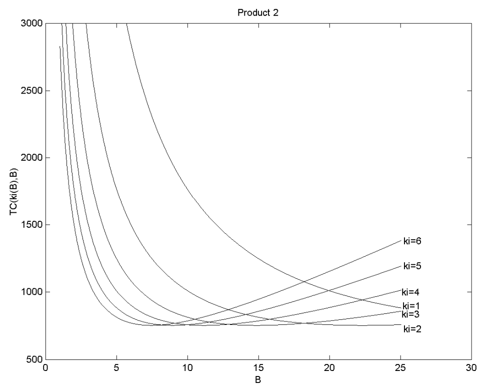


Figure 1 : The local minimum cost function of products 1 and 2

Remark 1 $\underline{TC}_i$ (B) function is piece-wise convex with respect to B (since it is easy to verify that $\frac{\partial^2 TC_i(B)}{\partial B^2} > 0$, for a given k_i.)

Remark 2 For each k_i, one can secure the minimum cost for product i, $\underline{TC}_i$ (k_i, B), at

$$B = \lambda_i(k_i) = \frac{1}{k_i}\sqrt{\frac{2a_i}{h_i d_i}} \tag{5}$$

with the minimum cost value of

$$\Upsilon_i = \min_B\{TC_i(B)\} = \sqrt{2a_i h_i d_i}\ . \tag{6}$$

The minimum cost Υ_i is exactly the *EOQ* foumula and Υ_i is, in fact, independent of k_i and B.

The following proposition shows the optimality structure of the *JRP*.

Proposition 1 The $TC_{GI}(B)$ function is piece-wise convex with respect to B.

Proof. The proof is presented in Appendix 6.1. ■

Again, we employ the data in Section 4 to demonstrate this comment in Figure 2. One may also observe that the term raises the left-tail of $TC_{GI}(B)$ function as the values of B decrease.

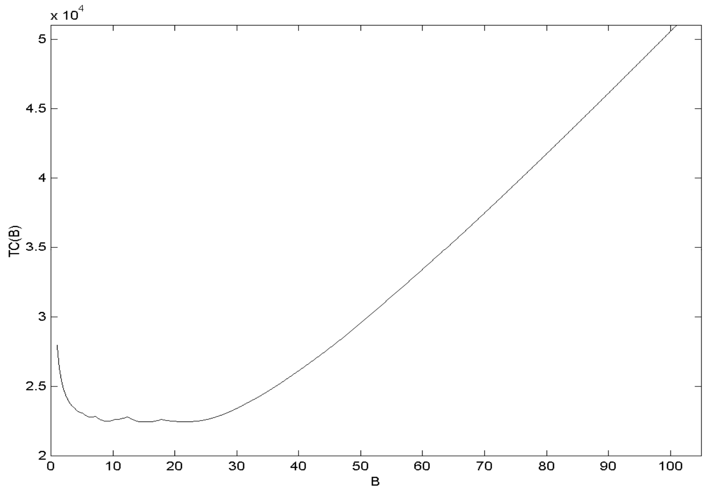


Figure 2: The optimal value for the $TC_{GI}(B)$ function.

2.2 The junction points on the optimal cost function

Next, we introduce the "junction points" on the curve of the TC_{GI} function. Recall that the $\underline{TC}_i\,(B)$ function is piece-wise convex. We define a junction point for the $\underline{TC}_i\,(B)$ function as a particular value of B where two consecutive convex curves concatenate. These junction points determine at 'what value of B' where one should change the multiplier of 'what product i' from k_i (= j) to k_i (= j+1) so as to secure the minimum value for $\underline{TC}_i\,(B)$ function.

By equation (3), we define the difference function $\Delta_i(k_i, B)$ by

$$\Delta_i(k_i, B) \equiv TC_i(k_i+1, B) - TC_{,i}(k_i, B) \qquad (7)$$

$$= -\frac{a_i}{k_i(k_i+1)B} + \frac{h_i d_i}{2}B. \qquad (8)$$

By Figure 1, one observes that the value of $\Delta_i(k_i, B)$ grows from negative to positive, and reaches zero at

$$\delta_i(k_i) = \sqrt{\frac{1}{k_i(k_i+1)}}\sqrt{\frac{2a_i}{h_i d_i}} \qquad (9)$$

where $\delta_i(k_i)$ is named as the *junction point* for the multipliers k_i and k_i +1 of product i. More specifically, $\delta_i(k_i = j)$ is the j^{th} junction point of product i where $j \in \aleph^+$. Therefore, the junction point $\delta_i(j)$ provides us the information that one should choose $k_i = j$ for $B \geq \delta_i(j)$ and choose $k_i = j+1$, *vice versa*, to secure a lower value for the $TC_i(B)$ function. In other words, if the value j is the optimal multiplier for $B \geq \delta_i(j)$, one should replace $k_i = j$ with $k_i = j+1$ as the optimal multiplier for product i at the junction point $\delta_i(j)$ if one searches from higher values to lower values of B.

Remark 3 For a product i,

$$\delta_i(k_i) = \sqrt{\frac{k_i}{k_i+1}}\;\lambda_i(k_i) \qquad (10)$$

prescribes the relation between a local minimum $\lambda_i(k_i)$ (defined in eq. 5) and the next junction point $\delta_i(k_i)$ (defined in eq. 9) below $\lambda_i(k_i)$.

2.3 The location of the junction points

By plugging k_i (using general-integer values) in equation (9) for all n products, one secures all the junction points. Again, we use the 10-product example in Section 4 to illustrate our discussion on the junction points. We sort all the junction points in descending order and list only those junction points that are less than T_{cc}^{*} =24.7009 in Table 1 where T_{cc}^{*} is expressed in eq. (11).

Table 1 in fact reveals an overview of our proposed search scheme. It shows that one can make use of difference calculation not only to locate all the junction points, but also to indicate 'which product *i*' should replace its optimal multiplier k_i by k_i +1. The following theoretical results on the junction points provide strengthen foundation for such a search scheme.

Lemma 1 Suppose that $k_i^{(L)}$ and $k_i^{(R)}$, respectively, are the optimal multipliers of the left-side and right-side convex curves with regard to a junction point in the plot of the $\underline{TC}_{,i}(B)$ function. Then, $k_i^{(L)} = k_i^{(R)} + 1$.
Proof: The proof is presented in Appendix 6.2. ■

Proposition 2 All the junction points for each individual product *i*, will be inherited by the $TC_{GI}(B)$ function. In other words, if *w* is a junction point for a product *i*, *w* must also show as a junction point on the piece-wise convex curve of the $TC_{GI}(B)$ function.
Proof: The proof is presented in Appendix 6.3. ■
Theorem 1 is an immediate result of Lemma 1 and Proposition 2.

Theorem 1 Suppose that $K^{(L)}$ and $K^{(R)}$, respectively, are the set of optimal multipliers for the left-side and right-side convex curves with regard to a junction point in the plot of the $TC_{GI}(B)$ function. Then, there is one and only one product *i* such that $k_i^{(L)} = k_i^{(R)}+1$.

From another point of view, Theorem 1 provides an important implication: namely, *the set of optimal multipliers for the $TC_{GI}(B)$ function is invariant between each pair of consecutive junction points.*

Table 1: The location of the junction points that are less than $\breve{B}_1$

Obs.	Where on B-axis	How to change k_i	Obs.	Where on B-axis	How to change k_i
0	24.4339	T_{cc}	23	12.3150	k_7: 1 ⇨ 2
1	24.2091	k_3: 2 ⇨ 3	24	11.9151	k_5: 6 ⇨ 7
2	22.7921	k_4: 6 ⇨ 7	25	11.8262	k_4: 12 ⇨ 13
3	22.5374	k_6: 2 ⇨ 3	26	11.4453	k_1: 2 ⇨ 3
4	22.2911	k_5: 3 ⇨ 4	27	10.9490	k_4: 13 ⇨ 14
5	22.1313	k_8: 1 ⇨ 2	28	10.8266	k_3: 5 ⇨ 6
6	19.8238	k_1: 1 ⇨ 2	29	10.3188	k_5: 7 ⇨ 8
7	19.7386	k_4: 7 ⇨ 8	30	10.2607	k_{10}: 2 ⇨ 3
8	18.1926	k_2: 2 ⇨ 3	31	10.1929	k_4: 14 ⇨ 15
9	17.7721	k_{10}: 1 ⇨ 2	32	10.0791	k_6: 5 ⇨ 6
10	17.4078	k_4: 8 ⇨ 9	33	9.9645	k_2: 4 ⇨ 5
11	17.2666	k_5: 4 ⇨ 5	34	9.5346	k_4: 15 ⇨ 16
12	17.1184	k_3: 3 ⇨ 4	35	9.1502	k_3: 6 ⇨ 7
13	15.9364	k_6: 3 ⇨ 4	36	9.1003	k_5: 8 ⇨ 9
14	15.5700	k_4: 9 ⇨ 10	37	9.0351	k_8: 3 ⇨ 4
15	15.3188	k_9: 1 ⇨ 2	38	8.9562	k_4: 16 ⇨ 17
16	14.0981	k_5: 5 ⇨ 6	39	8.8443	k_9: 2 ⇨ 3
17	14.0836	k_4: 10 ⇨ 11	40	8.5184	k_6: 6 ⇨ 7
18	13.2599	k_3: 4 ⇨ 5	41	8.4440	k_4: 17 ⇨ 18
19	12.8641	k_2: 3 ⇨ 4	42	8.1396	k_5: 9 ⇨ 10
20	12.8565	k_4: 11 ⇨ 12	43	8.1360	k_2: 5 ⇨ 6
21	12.7775	k_8: 2 ⇨ 3	44	8.0930	k_1: 3 ⇨ 4
22	12.3443	k_6: 4 ⇨ 5	45	7.9872	k_4: 18 ⇨ 19

2.4 Locate all the junction points

Before locating all the junction points, one needs to know how many junction points shall be located for each product. This question leads to the problem of finding an upper bound on k_i. In fact, the lower bound on the value of B determines the upper bound on k_i. Suppose that $\bar{v}_i$ is an upper bound on the value of k_i. Then we need to locate $\bar{v}_i$ junction points for product i.

2.4.1 Find an upper bound on k_i

Under *GI* policy, a simple upper bound on k_i can be derived from the *Common Cycle* (*CC*) approach and the independent solution (which is denoted by *IS*, and it is expressed in eq. 6).

Hanssmann (1962) proposed the *Common Cycle* (*CC*) approach to solve the Economic Lot Scheduling Problem (*ELSP*). Recall that the *CC* approach is a special case in which it assumes that T_i=T (or k_i=1) for all i, *i.e.*, all the products share the same cycle. Graves (1979) commented that one can consider the *JRP* as a special case of the *ELSP* where the major setup may be regarded as an additional product *0* with no demand or production requirements.

The optimal solution of the *CC* approach is a well-known upper bound on the objective function of the conventional *ELSP*. Therefore, we may use the cost of the *CC* approach, denoted by TC^{cc}, as an upper bound on the objective function of the *JRP*. Denote as $T_{cc}^{\ *}$ the optimal solution for the *CC* approach. Then, one may easily secure $T_{cc}^{\ *}$ by the following expression.

$$T_{cc}^{\ *} = \max\{\sqrt{\frac{2(A+\sum_{i=1}^{n} a_i)}{\sum_{i=1}^{n} h_i d_i}}\} \qquad (11)$$

Let $IS(n\text{-}\{i\})$ = $\sum_{j=1, j\neq i}^{n} \sqrt{2a_j h_j d_j}$ Then, an upper bound on the average cost of product i is obtained by $TC^{cc}\text{-}IS(n\text{-}\{i\})\text{-}\frac{A}{B}$, and we have

$$TC_i(k_i,B) = \frac{a_i}{k_i B} + \frac{h_i}{2} d_i k_i B \ \leq \ TC^{cc}\text{-}IS(n\text{-}\{i\})\text{-}\frac{A}{B} \ .$$

Thus, for a given B, an upper bound $v_i(B)$ on k_i is obtained by

$v_i(B)$=

$$\left\lceil \frac{TC^{cc} - IS(n-\{i\}) - \frac{A}{B} + \sqrt{\left(TC^{cc} - IS(n-\{i\}) - \frac{A}{B}\right)^2 - 2a_i h_i d_i}}{h_i d_i B} \right\rceil \qquad (12)$$

By plugging in (12) a lower bound value on B, denoted by B_{lb}, one secures an upper bound $\bar{v}_i$ on k_i.

The next task is to determine B_{lb}, a lower bound on B for the search range in the *JRP*. Many researchers addressed their efforts on narrowing the search range in the solution algorithms for the *JRP*. (Please refer to Section 1 for references.) van Eijs (1993) proposed that B_{lb} = $\frac{2A}{TC^U}$ where TC^U is an upper bound on the objective value of the *JRP*. van Eijs derived another upper bound on the objective value of the *JRP* other than TC^{cc}. Based on our experience, van Eijs' bound is usually looser than TC^{cc}, our upper bound by the *CC* approach. Therefore, we secure our lower bound on B by $B_{lb} = \frac{2A}{TC^U}$ and the upper bound $\bar{v}_i = v_i(\frac{2A}{TC^{cc}})$ on k_i is determined accordingly.

2.4.2 The junction points locating procedure

We utilize the theoretical results presented above and propose an efficient procedure, viz., the *Junction Point* (*JP*) *Locating Procedure*, to locate all the junction points of the TC_{GI} function as follows.

***The JP Locating Procedure*:**

```
for $i = 1, \ldots, n$
        Compute $\bar{v}_i$ on $k_i$ by eq. (12).
        Set $found = 0$ and $j = 0$.
        while $found = 0$
            $k_i = j$.
            Compute $\delta_i(k_i) = \sqrt{\frac{1}{k_i(k_i+1)}}\sqrt{\frac{2a_i}{h_i d_i}}$.
            (refer to equation (9) for details).
            $j = j+1$.
            if $j > v_i$, then found = 1;
        endwhile
endfor
```

Let $v_{max} \equiv max_i\{\bar{v}_i + 1\}$. Since the junction points are no more than nv_{max}, *i.e.*, $\sum_{i=1}^{n}\{\bar{v}_i + 1\} \le nv_{\max}$. Therefore, the complexity of the *JP Locating Procedure* is bounded by $O(n\ v_{max})$.

By substituting $k_i = 1, 2, 3, ..., 6$ into the eq. (9), we enumerate the computing results for the 10-product example in Section 4 in Table 2. One may compare Table 2 with the sorted sequence in Table 1. For example, when the search started from w_1=104.4466 where one should change $k_4 = 1$ to $k_4 = 2$ for product 4. The algorithm continues its search and changes $k_4 = 2$ to $k_4 = 3$ at the next junction point $B = 60.3023$.

2.5 The *K-GI* Search Procedure

In this section, we present an efficient procedure to secure the set of optimal multipliers at a given value of B, which is denoted as $K_{GI}(B)$. Proposition 2 provides an easier way to secure each $k_i \in K_{GI}(B)$ by

$$k_i(B) = \left\{ \begin{array}{ll} 1, & B > \sqrt{\frac{a_i}{h_i d_i}} \\ m, & \sqrt{\frac{1}{m(m+1)}}\sqrt{\frac{2a_i}{h_i d_i}} < B \le \sqrt{\frac{1}{m(m-1)}}\sqrt{\frac{2a_i}{h_i d_i}} \end{array} \right\} \tag{13}$$

Therefore, for any B, one can secure $K_{GI}(B)$ using the *K-GI Search Procedure* as follows.

The *K-PoT Search Procedure:*

```
for $i = 1, ..., n$
        Set $found = 0$ and $m = 1$.
        if $B > \sqrt{\frac{a_i}{h_i d_i}}$, then $found = 1$.
        while $found = 0$
            $m \leftarrow m+1$.
            if $B > \sqrt{\frac{1}{m(m+1)}}\sqrt{\frac{2a_i}{h_i d_i}}$
            , then $found = 1$.
        endwhile
        Set $k_i(B) = m$.
endfor
```

Table 2: The junction points for the TC_{GI} function in the example

Product i	$k_i = 6$	$k_i = 5$	$k_i = 4$	$k_i = 3$	$k_i = 2$	$k_i = 1$
1	4.3259	5.1185	6.2688	8.0930	11.4453	19.8238
2	6.8761	8.1360	9.9645	12.8641	18.1926	31.5104
3	9.1502	10.8266	13.2599	17.1184	24.2091	41.9314
4	22.7921	26.9680	33.0289	42.6401	60.3023	104.4466
5	11.9151	14.0981	17.2666	22.2911	31.5244	54.6019
6	8.5184	10.0791	12.3443	15.9364	22.5374	39.0360
7	2.6873	3.1797	3.8943	5.0276	7.1101	12.3150
8	4.8295	5.7143	6.9985	9.0351	12.7775	22.1313
9	3.3428	3.9553	4.8442	6.2539	8.8443	15.3188
10	3.8782	4.5887	5.6200	7.2554	10.2607	17.7721

3. A Global Optimum Search Algorithm

In this section, we present a search scheme which secures the global optimal solution for the *JRP*. The search scheme secures the global optimum since it is able to locate all the local minima (which, in turn, depends on the junction points) of the $TC_{GI}(B)$ function. Recall that $B_{lb} = \frac{2A}{TC^{cc}}$ is the lower bound on the value of B for the *JRP*. (Please refer to §2.4.1.) The overview of our search scheme is summarized as follows.

1. Secure all the junction points, and sort them in a descending order.
2. Utilize T_{cc}^* in eq. (11) as the first upper bound of the search range.
3. Secure the set of optimal multipliers for each convex interval in (B_{lb}, T_{cc}^*).
4. Utilize the derivative information on the $TC_{GI}(B)$ to further shorten the search range, and secure a pair of tighter upper and lower bounds of the search range, which are denoted by B_μ and B_λ, respectively.
5. Secure the local optimum (if it exists) for each convex interval in the search range $[B_\lambda, B_\mu]$.
6. A global optimal solution is secured by picking the best solution among all the local minima secured in $[B_\lambda, B_\mu]$.

We have detailed discussion on each step of our search scheme in the following subsections.

3.1 The *JP* Sorting Procedure

Recall that each junction point $\delta_i(k_i)$ provides the information that one should change the optimal multiplier of item i from k_i to k_i+1 at $\delta_i(k_i)$ to secure the optimal value for the $TC_i(B)$ function. We show the *JP Sorting Procedure* as follows.

The JP Sorting Procedure:

1. Input all the junction point $\{\delta_i(k_i)|i=1,\ldots,n\}$ of the $TC_{GI}(B)$ function (secured by the *JP Locating Procedure*).
2. Generate an array of ordered pairs by inputing the first element of each order pair is the location (*i.e.*, the value of B) of the junction point and the second element is the identity of the product i.
3. Use the location as the key field, and sort the ordered-pair array secure in Step 2 in descending order.

Denote by $\{w_j\}$ the sequence of junction points generated by the *JP Sorting Procedure* where $w_{j+1} < w_j$, $j = 1, 2, \cdots$. Another sequence of product indices, denoted by $\{\iota_j(w_j)\}$, is generated accordingly to correspond to the w_j's. We now have in hand an array of (sorted) pairs $\{(w_j, \iota_j(w_j))\}$.

Since the *JP Sorting Procedure* sorts all the junction points, of which there are at most nv_{max}, its complexity is bounded by $O(nv_{max} \log nv_{max})$.

3.2 The first upper bound

Recall that the search scheme needs to locate all the local minima of the $TC_{GI}(B)$ function by securing the local minimum candidate that exists in each convex interval. In order to reduce the run time of the search scheme, we need to shorten the range of the search scheme. This may be achieved by skipping those values of B where no local minimum exists.

First, Proposition 3 asserts that T_{cc}^* in eq. (11) can be used to secure an upper bound of the search range. Recall that T_{cc}^* is the optimal solution for the CC approach that uses the set of multipliers $\{k_0 = 1\} \cup \{k_i = 1\}_{i=1}^{n}$.

Proposition 3 For the $TC_{GI}(B)$ function, there exist no local minima for $B > T_{cc}^*$.

Proof. The proof is presented in Appendix 6.4. ■

By Proposition 3, the search scheme may skip the junction points in (T_{cc}^*,∞). Consequently, we may set T_{cc}^* as the first upper bound of the search range.

3.3 The optimal multipliers

Recall that the third step of the search scheme is to secure the set of optimal multipliers for each convex interval in $[B_{lb}, T_{cc}^*]$. This step may be done by:

1. Secure $K_{GI}(T_{cc}^*)$, *i.e.*, the set of optimal multipliers at T_{cc}^*, by the *K-GI Search Procedure* in §2.5 .
2. Starting from T_{cc}^*, we use the ordered-pair array $\{(w_j, \iota_j(w_j))\}$ to secure the set of optimal multipliers for each interval (w_{j+1}, w_j).

After securing $K_{GI}(T_{cc}^*)$, the algorithm searches in descending order toward lower values in the sequence $\{w_j\}$. At each junction point w_j, by Theorem 1, one should change *one and only one* multiplier by replacing $k_{\iota_j(w_j)}$ with $k_{\iota_j(w_j)}+1$ to update the set of optimal multipliers. Denote by $K(w_j)$ the set of optimal multipliers in the interval (w_{j+1}, w_j). Therefore, we secure $K(w_j)$ one by one by

$$K(w_j) \equiv (K(w_{j-1}) \setminus \{k_{\iota_j(w_j)}\}) \cup \{k_{\iota_j(w_j)}+1\} \quad (14)$$

for all $w_j \in (B_{lb}, B_\mu)$ where '\' denotes set subtraction. Recall that Theorem 1 implicates that the set of optimal multipliers for the $TC_{GI}(B)$ function is invariant in each convex interval (i.e., between a pair of consecutive junction points). Hence, this step actually secures the set of optimal multipliers for all the values of $B \in (B_{lb}, B_\mu^1)$.

3.4 The shortened search range

Next, we demonstrate how to utilize the derivative information on the $TC_{GI}(B)$ function to secure the upper and lower bounds of the search range, *i.e.*, B_μ and B_λ, respectively.

Again, we would like to shorten the range of the search scheme by skipping those values of B where no local minimum exists. Recall that we have secured a candidate $B_\mu^1 = T_{cc}^*$ for the upper bound of the search range in §3.2. Here, we hope to find another upper bound $B_\mu < B_\mu^1$ to further shorten the search range.

One may notice in Figure 2 that the $TC_{GI}(B)$ function raises rapidly as B decreases because of the term for the major setup cost, *i.e.*, $\frac{A}{B}$. Also, the slope (*i.e.*, the *first-order derivative*) of the left-tail of the $TC_{GI}(B)$ function keeps being *negative* upto a point where we denote it as B_λ. It implies that B_λ is the largest value of B where no local minima of the $TC_{GI}(B)$ function exist below B_λ. Surely, it shows us an opportunity to skip those values of $B \le B_\lambda$, and B_λ can be considered as the lower bound on the search range. Similarly, we hope to find the lowest value of B_μ where $TC_{GI}(B)$ function keeps being *positive* for $B \ge B_\mu$ and hence, no local minima of the $TC_{GI}(B)$ function exist for $B \ge B_\mu$.

In order to locate B_λ and B_μ, we will check the derivative information of the junction points from B_{lb} and B^1_μ, respectively. Given a set of $\{k_i\}$, the first derivative of the $TC_{GI}(B)$ function is given by

$$TC'_{GI}(\{k_i\},B) = \frac{-A}{B^2} + \sum_{i=1}^{n}\left(\frac{-a_i}{k_i B^2} + \frac{h_i d_i k_i}{2}\right). \quad (15)$$

In the third step of the search scheme, we have secured the set of optimal multipliers for each convex interval $(w_{j+1}, w_j) \in (B_{lb}, B^1_\mu)$. We are ready to plug in $\{k_i\}$ in eq. (15) and check the value of the first derivative by the *Bounds Locating Procedure* as follows.

***The Bounds Locating Procedure*:**

```
Set $lb\_found = 0$, and $\hat{j} = \arg\min\{j \mid w_j \ge B_{lb}\}$.
Set $j = \hat{j} - 1$ and $B_\lambda = B_{lb}$.
while $lb\_found = 0$
    if $TC'_{GI}(K(w_j), w_j) > 0$,
        Set $\breve{j} = j$, $B_\lambda = \breve{B}(K(w_{\breve{j}}))$ and $lb\_found = 1$.
    otherwise, set $j = j - 1$.
endwhile
Set $lb\_found = 0$, and $\hat{j} = \arg\max\{j \mid w_j \le B^1_u\}$.
Set $j = \hat{j} - 1$ and $B_\mu = B^1_\mu$.
while $ub\_found = 0$
    if $TC'_{GI}(K(w_j), w_j) < 0$,
        Set $\breve{j} = j$, $B_\lambda = \breve{B}(K(w_{\breve{j}}))$ and $lb\_found = 1$.
    otherwise, set $j = j - 1$.
endwhile
Output $B_\lambda$ and $B_\mu$, stop.
```

The rationale behind the *Bounds Locating Procedure* is as follows. The variables w_j and w_{j-1} are consecutive junction points on the $TC_{GI}(B)$ function. By Theorem 1, the set of optimal multipliers for the $TC_{GI}(B)$ function is invariant between w_j and w_{j-1}. Therefore, w_j and w_{j-1} correspond to the *left-hand* and the *right-hand* end points for a particular piece of *convex* curve on the $TC_{GI}(B)$. For locating the lower bound B_λ, the condition $TC'_{GI}(K(w_j), w_j) > 0$ checks the lowest junction point where the sign of the first derivative of the $TC_{GI}(B)$ function changes. The value of $\breve{B}(K(w_{\breve{j}}))$ indicates the lowest-valued local minimum for the $TC_{GI}(B)$ function. We set the lower bound of the search range B_λ at $\breve{B}(K(w_{\breve{j}}))$ since no local minima of the $TC_{GI}(B)$ function exist below $B_\lambda = \breve{B}(K(w_{\breve{j}}))$. Similar idea applies to locate the upper bound B_μ, but the condition checks $TC'_{GI}(K(w_j), w_j) < 0$ for the largest junction point.

3.5 Secure all the local optima

In this section, we introduce the condition for checking the existence of a local optimum in the interval of (w_{j+1}, w_j).

In the third step of the search scheme, we already secure the set of optimal multipliers for all the values of $B\in$ (B_{lb}, B_μ). We are ready to plug in the set of optimal multipliers $\{k_i\}$ and check the existence of a local optimum $\breve{B}_j$ in the interval (w_{j+1}, w_j) by the *Location checking condition* as follows.

Location checking condition:

i. For the set of $\{k_i\}$, secure its local minimum at $\breve{B}_j(\{k_i\})$ by eq. (19).

ii. If $\breve{B}_j \in (w_{j+1}, w_j)$, then $\breve{B}_j$ is a local minimum of the $TC_{GI}(B)$ function.

3.6 The algorithm

We are now ready to enunciate the Global Optimum Search Algorithm. It uses the array of the (sorted) ordered pairs $\{(w_j,\ \ _j(w_j))\}$ as the backbone and secures all the local minima of the $TC_{GI}(B)$ function. Recall that the algorithm searches from the upper bound B_μ along lower value we label as l the index for the local optima of the $TC_{GI}(B)$ curve. Hence, $\breve{B}_l$ is the l^{th} local optimal solution secured in the search process of the Global Optimum Search algorithm. The step-by-step procedure is presented as follows.

1. Generate the array of the (sorted) ordered pairs, *i.e.*, $\{(w_j, \iota_j(w_j))\}$, by
   (a) Secure all the junction points of TC_{GI} function by the *JP Locating Procedure*.
   (b) Sort all the junction points by the *JP Sorting Procedure*.
2. Utilize T_{cc}^* in eq. (11) to secure B_μ^1 by:

(a) Set $K_{GI}(w_1+\varepsilon)$ = {1, 1,…, 1}. Employ the Location checking condition in §3.5 to check:

   i. If $\breve{B}(K_{GI}(w_1+\varepsilon)) = T_{cc}^* \in (w_1, \infty)$, then set l = 1, $\breve{B}_1$ = T_{cc}^*, and compute $TC_{GI}(K_{GI}(w_1+\varepsilon), T_{cc}^*)$; go to Step 2(*b*).

   ii. otherwise, let l = 0, go to Step 2(*b*).

(b) Set B_μ^1 = T_{cc}^* and $\hat{j} = \arg\max\{w_i \le T_{cc}^*\}$. Do the following steps:

   i. Secure $K(T_{cc}^*)$, *i.e.*, the set of optimal multipliers at T_{cc}^* by the *K-GI Search Procedure* in § 2.5.

   ii. Set $K(w_j) \equiv (K(T_{cc}^*) \setminus \{k_{l_j(w_j)}\}) \cup \{k_{l_j(w_j)}+1\}$.

3. Secure the set of optimal multipliers for the convex intervals (w_{j+1}, w_j) $\in$ (B_{lb}, B_μ^1) by:
   (a) If $w_j < B_{lb}$, then go to Step 4.
   (b) Otherwise, let $j = j$+1, secure $K(w_j)$ by
   $$K(w_j) \equiv (K(w_{j-1}) \setminus \{k_{l_j(w_j)}\}) \cup \{k_{l_j(w_j)}+1\}$$
   and go to Step 3(*a*).
4. Use the *Bounds Locating Procedure* in §3.4 to secure the upper and lower bounds, *i.e.*, B_μ and B_λ, respectively.
5. Secure the local optimum (if it exists) for each interval (w_{j+1}, w_j)$\in$ [B_λ, B_μ] by:
   (a) Locate the junction point $j = \arg\max\{w_i \le B_u\}$.
   (b) Employ the Location checking condition in §3.5 to check: if $\breve{B}(K(w_{\bar{j}})) \in (w_{j+1}$, $w_j)$, then let l= l+1, $\breve{B}_l$ = $\breve{B}(\tilde{K}(w_j))$, and compute TC_{GI} ($\tilde{K}(w_j)$, $\breve{B}_l$).
   (c) Set $j = j$ +1 and check: if $w_j < B_\lambda$, then go to Step 6; otherwise, go to Step 5(*b*).
6. Secure the global optimal solution by
$$(K_{GI}^*, B_{GI}^*) = \arg\min_l \{TC_{GI}(K(w_j), \breve{B}_l)\}. \quad (16)$$

and stop.

Recall that the complexity of the *JP Locating Procedure* and the *JP Sorting Procedure* is bounded by $O(nv_{max})$ and $O(nv_{max}\log nv_{max})$, respectively. The complexity of Steps 3 and 4 is also bounded by $O(nv_{max})$. The number of iterations in the loop of Step5 is less than $\sum_i (v_i + 1)$, and is surely less than nv_{max}. Therefore, the complexity of the global optimum search algorithm is bounded by $O(nv_{max}\log nv_{maxx})$.

4. A Numerical Example

In this section, we present a numerical example to demonstrate the implementation of the proposed Global Optimum Search algorithm. In the Table 3, we present the set of parameters used in this numerical example. Part of the data also show in Example 6 of Elmaghraby (1978).

Table 3: The set of parameters used in this numerical example

Product	1	2	3	4	5	6	7	8	9	10
Minor setup cost	33600	16800	4800	7200	14400	24000	72000	14400	13200	84000
Holding cost	0.095	0.0235	0.0065	0.022	0.023	0.075	0.1055	0.014	0.0625	0.2955
Demand	900	720	420	30	210	210	4500	2100	900	900
Major setup cost	6250									

1. Secure all the junction points of TC_{GI} function by the *JP Locating Procedure* as shown in Table 1. Then, we sort all the junction points by the *JP Sorting Procedure*.
2. A candidate for the upper bound of the search range is secured by $B_\mu^1 = T_{cc}^* =24.7009$.
   (a) We note that $T_{cc}^* = 24.7009 \notin (w_1, \infty) = (104.4466, \infty)$.
   (b) The largest junction point less than B_μ^1 is secured at w_{11}=24.091. Also,
      i. The set of optimal multipliers at $T_{cc}^*, K(T_{cc}^*)$, is secured by {1,2,2,6,3,2,1,1,1,1}.
      ii. Set $K(w_{11}) \equiv (K(T_{cc}^*) \setminus \{k_{l_{11}(w_{11})}\}) \cup \{k_{l_{11}(w_{11})} + 1\}$ which is given by {1,2,3,4,3,2,1,1,1,1}.
3. Secure the set of optimal multipliers for the convex intervals $(w_{j+1}, w_j) \in (B_{lb}, B_\mu^1) =$ (0.5312, 24.7009). In this step, we totally review 935 convex intervals.
4. Use the *Bounds Locating Procedure* in §3.4 to secure the upper and lower bounds. They are given by B_λ =5.0078 (the 92nd junction point) and B_μ =22.1313 (the 15th junction point), respectively.
5. Secure the local optimum (if it exists) for each interval $(w_{j+1}, w_j) \in [B_\lambda, B_\mu]$. All the local minima secured are summarized in Table 4.
6. A global optimal solution is secured at B_{GI}^* =14.9114 and K_{GI}^* is given by {2,3,4,10,5,4,1,2,2,2}. The optimal total cost is given by $22432.46.

Originally, we have to search totally 946 junction points. By securing a candidate on the search range, namely T_{cc}^*, we can skip (the largest) 10 junction points. Furthermore, by utilizing the derivative information, we shorten the search range from (0.5312, 24.7009) to (5.0078, 22.1313). This action leads to only 78 *possible* local minima to be checked. Consequently, it reduces significantly the run time for the search scheme.

Table 4: The Search Process of the Global Optimum Search Algorithm

(w_{j+1},w_j)	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	$\breve{B}_l$	$TC_{PoT}(\breve{B}_l)$
[19.8238, 22.1313]	1	1	2	3	7	4	3	1	2	1	1	21.2856	22445.06
[19.7386, 19.8238]	1	2	2	3	7	4	3	1	2	1	1	19.7388	22501.68
[18.1926, 19.7386]	1	2	2	3	8	4	3	1	2	1	1	19.7274	22501.68
[15.3188, 15.5700]	1	2	3	4	10	5	4	1	2	1	2	15.4952	22439.14
[14.0981, 15.3188]	1	2	3	4	10	5	4	1	2	2	2	14.9114	22432.46*
[10.3188, 10.8266]	1	3	4	6	14	7	5	2	3	2	2	10.5447	22612.63
[9.1502, 9.5346]	1	3	5	6	16	8	6	2	3	2	3	9.1826	22490.33
[6.5060, 6.5665]	1	4	7	9	23	12	8	3	5	3	4	6.5305	22778.11
[6.2869, 6.5060]	1	4	7	9	23	12	9	3	5	3	4	6.5012	22778.26
[6.2508, 6.2539]	1	5	7	9	24	12	9	3	5	4	4	6.2516	22794.09
[6.1824, 6.2508]	1	5	7	10	24	12	9	3	5	4	4	6.2469	22794.09
[5.0078, 5.0276]	1	6	9	12	29	15	11	4	6	4	5	5.0258	23083.62

5. Concluding Remarks

This paper fulfills two research gaps in the study of the Joint Replenishment Problem (*JRP*). First, our study presents several important results on the optimality structure of the *JRP* under General Integer (*GI*) policy. For instance, Proposition 1 asserts that $TC_{GI}(B)$ function is a *piece-wise convex* function of *B*. Also, we have thorough discussion on the properties of the junction points on the $TC_{GI}(B)$ function in Section 2.

Second, we propose an efficient search algorithm that always secures the global optimal solution for the *JRP* under *GI* policy in Section 3. In our search algorithm, we use simple, but powerful bounds that significantly shorten the search range. Also, we utilize the junction points on the $TC_{GI}(B)$ function, to efficiently secure all the local minima in the search range.

Our search algorithm is the first solution approach in literature that always secures the global optimal solution though many heuristics have been derived for the *JRP*. Also, the theoretical results in this paper shall establish an important foundation for those lot sizing and scheduling problems.

Acknowledgement The authors would like to thank the National Science Council, Taiwan, Republic of China for its support (grant NSC-89-2213-E-029-029).

6. Appendix

6.1 Proof for Proposition 1

Proof. The first term in the $TC_{GI}(B)$ function, *i.e.*, $\frac{A}{k_0 B}$, is a convex function. The second term, *i.e.*, $\sum_{i=1}^{n} \underline{TC}_i(B)$, is a piece-wise convex function since it is the sum of *n* piece-wise convex functions by Remark 1. Since the $TC_{GI}(B)$ function is the sum of a convex function and a piece-wise convex function, it is obvious piece-wise convex. ■

6.2 Proof for Lemma 1

Proof. By equation (9),

$$\delta_i(\bar{v}_i) < ... < \delta_i(j+1) < \delta_i(j) < ... < \delta_i(2) < \delta_i(1) \qquad (17)$$

where $\bar{v}_i$ is an upper bound on k_i (discussed in § 2.4). Denote as k_i^* (*B*) the optimal multiplier for $\underline{TC}_i(B)$ at a given *B*. Because of ineq. (17) and the convexity of $TC_i(k_i, B)$, one asserts that

$$k_i(B) = \begin{cases} 1, & if \ B \in [\delta_i(1), \infty) \\ j+1, & if \ B \in [\delta_i(j+1), \delta_i(j)), \ for \ j = 1, ..., \bar{v}_i - 1 \end{cases} \qquad (18)$$

Equation (18) exactly states that $k_i^{(L)} = k_i^{(R)} + 1$. ■

6.1 Proof for Proposition 2

Proof. Recall that the function TC_{GI} is a separable function where $TC_{GI}(B) = \frac{A}{k_0 B} + \sum_{i=1}^{n} \underline{TC}_i(B)$ where $k_0 = 1$. Without loss of generality, assume that *w* is a junction point for a product *i*, but *not* a junction point for the other (*n*-1) products. Then, there must exist $\varepsilon > 0$ such that the followings hold.

1. the curve for $\sum_{j \neq i} \underline{TC}_j(B)$ is convex in the interval of $[w-\varepsilon, w+\varepsilon]$ since each one of $\underline{TC}_j(B)$ is convex in $[w-\varepsilon, w+\varepsilon]$ where $j \neq i$,
2. $\underline{TC}_i(B)$ is convex in the intervals of $[w-\varepsilon, w]$ and $[w, w+\varepsilon]$.
3. $\frac{A}{k_0 B}$ is convex in the intervals of $[w-\varepsilon, w]$ and $[w, w+\varepsilon]$.

Since $TC_{GI}(B) = \frac{A}{k_0 B} + \underline{TC}_i(B) + \sum_{j \neq i} \underline{TC}_j(B)$, $TC_{GI}(B)$ is still convex in the intervals $[w-\varepsilon, w]$ and $[w, w+\varepsilon]$. Therefore, *w* is a junction point on the curve of $TC_{GI}(B)$. ■

6.4 Proof for Proposition 3

Proof. For any given set of $\{k_i\}$, one may secure its local minimum, $\breve{B}(\{k_i\})$, by (1) securing the derivative of the $TC_{GI}(B)$ function *w.r.t.* *B*, and (2) equating it to zero. $\breve{B}(\{k_i\})$ is given by eq. (19) as follows.

$$\breve{B}(\{k_i\}) = \sqrt{\frac{2(A + \sum_{i=1}^{n} \frac{a_i}{k_i})}{\sum_{i=1}^{n} (h_i d_i k_i)}} \ . \qquad (19)$$

It is obvious that $\breve{B}(\{k_i\}) \leq T_{cc}^*$ since $k_i \geq 1$ for all *i*. Therefore, there exists no local minima for $B > T_{cc}^*$.■

References

1. Aksoy, Y. and Erenguc, S. (1988), “Multi-item inventory models with coordinated replenishments: A survey”, *International Journal of Production Management*, Vol. 8, pp. 63-73.
2. Arkin, E., Joneja D., Roundy, R. (1989), “Computational complexity of uncapacitated multi-echelon production planning problems”, *Operations Research Letters*, Vol. 8, pp. 61-66.
3. Elmaghraby, S.E. (1978), “The economic lot scheduling problem (ELSP): review and extension”, *Management Science*, Vol. 24, pp. 587-597.
4. Goyal, S.K. (1973a), “ Determination of economic packaging frequency for items jointly replenished”, *Management Science*, Vol. 20, No. 2, pp. 232-235.
5. Goyal, S.K. (1973b), “Economic packaging frequency for Items jointly replenished”, *Operations Research*, Vol. 21, pp. 644-647.

6. Goyal, S.K. (1974a), “Optimum ordering policy for a multi item single supplier system”, *Operational Research Quarterly*, Vol. 25, No. 2, pp. 293-298.
7. Goyal, S.K. (1974b), “Determination of optimum packaging frequency of items jointly replenished”, *Management Science*, Vol. 21, No. 4, pp. 436-443.
8. Goyal, S.K. (1988), “Determining the optimal production-packaging policy for jointly replenished items”, *Engineering Costs and Production Economics*, Vol. 15, pp. 339-341.
9. Goyal, S.K. and Belton, A.S. (1979), “On a simple method of determining order quantities in joint replenishments under deterministic demand”, *Management Science*, Vol. 26, Iss. 6, pp.604.
10. Goyal, S.K., Satir, A.T. (1989), “Joint replenishment inventory control: deterministic and stochastic models”, *European Journal of Operational Research*, Vol. 38, pp. 2-13.
11. Graves, S.C., (1979), “On the deterministic demand multi-product single-machine lot scheduling problem”, *Management Science*, Vol. 25, No. 3, pp. 276-280.
12. Hanssmann, F., *Operations Research in Production and Inventory* (New York City: Johnson Wiley & Sons), pp. 158-160 (1962).
13. Jackson, P., Maxwell, W. and Muckstadt, J. (1985), “The joint replenishment problem with a powers-of-two restriction”, *IIE Transactions*, Vol. 17, No. 1, pp. 25-32.
14. Kaspi, M. and Rosenblatt, M.J. (1983), “An improvement of silver’s algorithm for the joint replenishment problem”, *IIE Transactions*, Vol. 15, pp. 264.
15. Nocturne, D.J. (1973), “Economic ordering frequency for several items jointly replenished”, *Management Science*, Vol. 19, No. 9, pp. 1073-1099.
16. Shu, F.T. (1971), “Economic ordering frequency for two items jointly replenished”, *Management Science*, Vol. 17, No. 6, pp. B406-B410.
17. Silver, E.A. (1976), “A simple method of determining order quantities in jointly replenishments under deterministic demand”, *Management Science*, Vol. 22, No. 12, pp. 1351-1361.
18. Stevenson, and William. J. (1993), *Production/Operations Management*, 4th ed., Homewood, IL: Irwin.
19. van Eijs, M.J.G. (1993), “A note on the joint replenishment problem under constant demand”, *Journal of Operational Research Societ*y, Vol. 44, pp. 185-191.
20. Viswanathan, S. (1996), “A new optimal algorithm for the joint replenishment problem”, *Journal of Operational Research Society*, Vol. 47, pp. 936-944.
21. Wildeman, R.E., Frenk, J.B.G., Dekker, R. (1997), An efficient optimal solution method for the joint replenishment problem. *European Journal of Operational Research*, Vol. 99, 433-444.