Journal of the Korean Institute of Industrial Engineers
Vol. 29, No. 3, pp. 222-229, September 2003.

# 2 지역/지정위치 저장시스템의 분석과 최적화

양 문희†

단국대학교 공학부(산업공학 전공)

# Analysis and Optimization of a 2-Class-based Dedicated Storage System

Moonhee Yang

Department of Industrial Engineering, Dankook University, Cheonan, 330-714

In this paper, we address a layout design problem, PTN[2], for determining an appropriate 2-class-based dedicated storage layout in a class of unit load storage systems. Our strong conjecture is that PTN[2] is NP-hard. Restricting PTN[2], we provide three solvable cases of PTN[2] in which an optimal solution to the solvable cases is one of the partitions based on the PAI(product activity index)-nonincreasing ordering. However, we show with a counterexample that a solution based on the PAI-nonincreasing ordering does not always give an optimal solution to PTN[2]. Utilizing the derived properties, we construct an effective heuristic algorithm for solving PTN[2] based on a PAI-nonincreasing ordering with performance ratio bound. Our algorithm with $O(n^2)$ is effective in the sense that it guarantees a better class-based storage layout than a randomized storage layout in terms of the expected single command travel time.

***Keywords***: class-based dedicated storage layout, unit load system, AS/RS

## 1. Introduction

A unit load can be defined as a unit to be moved or handled at one time. A storage system can be called a unit load storage system where unit loads are stored, handled, and retrieved. Automated storage/retrieval systems (AS/RS) or rack-supported storage systems can be the type of unit load systems (Yang, 1988). K-class-based dedicated storage policy or simply K-class-based storage policy employs K zones in which lots from a class of products, are stored randomly. Tompkins and White(1984) pointed out that class-based storage with randomized storage within each class can yield both the throughput benefits of dedicated storage and the space benefits of randomized storage. Also they suggested that in order to achieve both benefits, three to five classes may be defined.

There have appeared many papers such as Cho (2001), Lee(1998), Bozer(1998), Chang(1995), and Hausman(1976) so on, which focused on both benefits or either the throughput benefits or the space benefits based on simulation techniques under some operating policies.

In this paper, based on combinatorics, we define a deterministic 2-class-based dedicated storage problem in a unit load system and provide an effective heuristic algorithm in addition to basic theoretical results of 2-class-based dedicated storage policy. We make "the constant-space assumption" that the number of storage locations for a class is not the maximum aggregate inventory position for a class but the sum of space requirement for products assigned to the class. In fact, the constant-space assumption is made since the problem for minimizing the maximum aggregate

The Research was conducted by the research fund of Dankook University in 2003.
† Corresponding author : Professor Moonhee Yang, Department of Industrial Engineering, Dankook University, San 29 Anseo-Dong Cheonan, Choongnam, 330-714 Korea, Fax : +82-041-550-3570, e-mail : myfriend@dankook.ac.kr
Received March 2003; revision received July 2003; accepted July 2003.

inventory position is well known to be NP-hard. In addition, our strong conjecture is that our 2-class-based dedicated storage problem seems to be NP-hard even if it is made.

In section 2, we introduce a 2-class-based dedicated storage problem denoted by PTN[2]. In section 3, since our strong conjecture is that PTN[2] is NP-hard, we provide three solvable cases by relaxing PTN[2]. We prove that an optimal solution to the solvable cases is based on a PAI-nonincreasing ordering. Especially, we prove that an optimal solution to a solvable case denoted by PTL[2] is based on a PAI-nonincreasing ordering and provide a greedy algorithm with O(n). In section 4, we give a counter- example in order to show that a solution based on a PAI-nonincreasing ordering does not always give an optimal solution to PTN[2]. In section 5, an effective heuristic algorithm with $O(n^2)$, denoted by ALGPTN [2], for solving PTN[2] is constructed based on a PAI-nonincreasing ordering since the PAI indexes still an effective approach to solving PTN[2] in the sense that it guarantees a better class-based storage layout than a randomized storage layout in terms of the expected SC travel time. In addition, some properties for PTN[2] are analyzed as well as a performance ratio bound.

For convenience to reader, the list of symbols used in this paper is given in <Table 1>. To denote optimality for a decision variable, a superscript (*) will be used at the upper right side of each symbol.

## 2. 2-Class-Based Storage Problem

Our storage system consists of R storage locations each of which accommodates only one unit load. The storage/retrieval operation is based on the 2-zone-based storage policy and within each zone, a storage location is equally likely to be selected for a storage operation, i. e., random assignment rule (RAN rule) is used.

The expected one-way travel time from a Pick-up/ Deposit (P/D) station to storage location j is given as $t_j$ for j=1, 2, ..., R. Without loss of generality, it is

**Table 1.** Notation List

| Notation | Meaning |
|---|---|
| $A_k$ | set of storage locations assigned to zone k |
| $C_k$ | a set of products assigned to class k |
| $CAI_k$ | $= \frac{D_k}{R_k}$, class activity index of class k |
| D | $= \sum_{k=1}^{K} D_k = \sum_{i=1}^{n} d_i$ |
| $d_i$ | average retrieval rate of product i, for i=1, ..., n |
| $D_k$ | $= \sum_{i \in C_k} d_i$, average retrieval rate from class k |
| $E(SC_K)$ | expected SC travel time given K classes |
| $E(SC_K \mid LAY)$ | expected SC travel time given LAY |
| $E(SC_K \mid P)$ | expected SC travel time given P |
| K | number of classes or zones used in a unit load system |
| LAY | $= \{A_1, A_2, \cdots, A_K\}$, a layout given K zones |
| n | number of products |
| P | $= \{C_1, C_2, \cdots, C_K\}$, a partition given K classes |
| $PAI_i$ | $= \frac{d_i}{r_i}$, product activity index of product i, for i=1, ..., n |
| $r_i$ | space requirement of product i when it is replenished |
| R | $= \sum_{k=1}^{K} R_k = \sum_{i=1}^{n} r_i$ |
| $R_k$ | $= \lvert A_k \rvert$, number of storage locations required for zone k |
| $t_j$ | one-way travel time to storage location j |
| $T_k$ | expected SC travel time from an i/o point to zone k |

assumed that $t_1 \le t_2 \le \cdots \le t_R$. Let $A_k$ be a set of storage locations assigned to zone k for k=1, 2. We assign the first $|A_1|$ storage locations to $A_1$ based on the $t_j$-nondecreasing ordering and the remaining storage locations to $A_2$ where $|X|$ denotes the cardinality of set X. It follows that $A_1 = \{1, 2, \cdots, |A_1|\}$ and $A_2 = \{|A_1|+1, |A_1|+2, \cdots, R\}$.

An arriving replenishment lot of a product i, the size of which is $r_i$ in unit load, contains a single product and are assigned randomly to open storage locations in one of two separate zones by using an storage/retrieval (S/R) machine or operator which or who can carry only one unit load at a time. Let $C_k$ be the set (or class) of products assigned to zone k. Then space requirement or the number of storage locations required for class k, $R_k$, can be expressed as

$$R_k = |A_k| = \sum_{j \in C_k} r_j \tag{1}$$

The average demand rate for a product i, $d_i$ unit loads/unit time, which is defined as the average number of retrievals per unit time, is given as a real constant in advance. Retrievals are performed on first-in first-out basis. The average demand rate from zone k, $D_k$ is obtained as $\sum_{i \in C_k} d_i$. Since practically a class contains at least one product, it can be assumed that $|C_k| \ge 1$.

Our objective is to minimize $E(SC_2)$, the expected single command travel time as follows. The expected SC travel time to zone k, $T_k$, can be expressed as

$$T_k = \frac{2}{|A_k|} \sum_{j \in A_k} t_j \tag{2}$$

Since the probability of visiting zone k is $\frac{D_k}{D}$, $E(SC_2)$ can be expressed as equation (4) by replacing $T_k$ in equation (3) with equation (2).

$$E(SC_2) = \sum_{k=1}^{2} \frac{D_k}{D} T_k \tag{3}$$

$$E(SC_2) = \frac{2}{D} \sum_{k=1}^{2} \frac{D_k}{|A_k|} \sum_{j \in A_k} t_j \tag{4}$$

where $D = D_1 + D_2$. Hence our problem can be described as

PTN[2] : Given n products with $\{(r_i, d_i), i = 1, 2, \cdots, n\}$, $\{t_j, j = 1, 2, \cdots, M\}$, find an optimal partition, $P^* = \{C_1^*, C_2^*\}$ such that we minimize

Minimize $\{(r_i, d_i), i = 1, 2, \cdots, n\}$

$$Z(P) = E(SC_2 \mid P) = \frac{2}{D} \sum_{k=1}^{2} \frac{D_k}{R_k} \sum_{j \in A_k} t_j$$

subject to $|C_k| \ge 1$ for k=1, 2

$$D_k = \sum_{i \in C_k} d_i$$

$$R_k = |A_k| = \sum_{i \in C_k} r_i$$

## 3. Solvable Cases of PTN[2]

Since our strong conjecture is that PTN[2] is NP-hard, we will provide some special solvable cases of PTN[2] by restricting PTN[2]. Define the activity index of product i ( $PAI_i$) as $\frac{d_i}{r_i}$.

### 3.1 Restriction of travel time : $t_j$ = pj+q

Proposition 1. If $t_j$ = pj+q for all j, then $P^*$ is one of the partitions based on a PAI- nonincreasing ordering.

Proof : If we replace $t_j$ with (pj+q), equation (2) can be reduced to

$$T_1 = p(R_1 + 1) + 2q \tag{5}$$

$$T_2 = p(2R_1 + R_2 + 1) + 2q \tag{6}$$

Replacing $T_1$, $T_2$ of equation (4) with equation (5) and equation (6), we have,

$$E(SC_2) = \frac{D_1}{D} T_1 + \frac{D_2}{D} T_2 = \frac{p}{D}\{D + DR + (R_1 D - D_1 R)\} + 2q \tag{7}$$

Let $x_i$=1 if product i is assigned to zone 1 and $x_i$=0 otherwise. Let $c_i = r_i D - d_i R$. Since $R_1 = \sum_{i=1}^{n} r_i x_i$ and $D_1 = \sum_{i=1}^{n} d_i x_i$, $R_1 D - D_1 R = \sum_{i=1}^{n} c_i x_i$. Hence equation (7) can be further reduced to

$$E(SC_2) = \frac{p}{D}\left(D + DR + \sum_{i=1}^{n} c_i x_i\right) + 2q \tag{8}$$

Since D, R, p and q are constant, and each class must contain at least one product, the relaxed PTN[2] can be formulated as

PTL[2] : Minimize $Z = \sum_{i=1}^{n} c_i x_i$

subject to $\sum_{i=1}^{n} x_i \le$ (n-1)

$$x_i \in \{0, 1\}$$

It can be observed that (i) PTL[2] is a simple binary knapsack problem, (ii) the solution to PTL[2] is independent of $\{t_j\}$, and (iii) Z is minimized if we assign product i with negative $c_i$ to class 1 and assign product i with positive $c_i$ to class 2. Since $c_i = r_i D - d_i R < 0$ if and only if $PAI_i > \frac{D}{R}$, an optimal solution to PTL[2] can be obtained by taking the products by PAI-nonincreasing order and assigning the first m products to class 1 where m is an integer such that $\frac{d_m}{r_m} \geq \frac{D}{R}$. Note that if $\frac{d_i}{r_i} = \frac{D}{R}$, the product i can be assigned to either class 1 or class 2. Hence, $P^*$ is one of the partitions based on a PAI- nonincreasing ordering.

From Proposition 1, the greedy algorithm, which solves PTL[2], can be summarized as follows:

ALGPTL[2]
Step 1. Compute v= $\frac{D}{R}$.
Step 2. For i=1 to n, do
Begin
If $PAI_i \geq$ v, the assign product i to $C_1^*$.
otherwise, assign product i to $C_2^*$
End

Proposition 2 ALGPTL[2] solves PTL[2] in O(n).

Proof: As shown in ALGPTL[2], Step 1 and Step 2 each requires O(n). Since $P^*$ is an optimal solution to PTL[2] if and only if $P^*$ satisfies $PAI_i > \frac{D}{R} \geq PAI_j$ for $i \in C_1^*$, $j \in C_2^*$, ALGPTL[2] solves PTL[2] in O(n).

## 3.2 Restriction of space requirement : $r_i = r$

Proposition 3. If $r_i = r$ for all i, then $P^*$ is one of the partitions based on a PAI-nonincreasing ordering.

Proof: Consider a partition, P = $\{C_1, C_2\}$. Choose product s from $C_1$ and product t from $C_2$ such that

$$d_s = \min_{i \in C_1}(d_i) \text{ and } d_t = \max_{i \in C_2}(d_i) \quad (9)$$

Define $\delta = d_t - d_s$. Let P' be a partition resulted from swapping product s and t. Since $r_i = r$ for all i, using equation (3), we have

$$E(SC_2 \mid P) = \frac{1}{D}(D_1T_1 + D_2T_2) \quad (10)$$

$$E(SC_2 \mid P') = \frac{1}{D}\{(D_1+\delta)T_1 + (D_2-\delta)T_2\} \quad (11)$$

Subtracting equation (11) from equation (10) gives

$$v = E(SC_2 \mid P) - E(SC_2 \mid P') = \frac{\delta}{D}(T_2 - T_1) \quad (12)$$

Since $(T_2 - T_1) \geq 0$, $v \geq 0$ if and only if $\delta \geq 0$. Hence if $d_t \geq d_s$, then swapping two products s and t does not increase the expected SC travel time. In the similar manner, continuing to swap two products satisfying both equation (9) and $\delta \geq 0$ results in a $d_i$ (or PAI)-nonincreasing ordering eventually.

## 3.3 Restriction of retrieval rate : $d_i = d$

Proposition 4. If $d_i = d$ for all i, then $P^*$ is one of the partition based on a PAI-nonincreasing ordering.

Proof: Consider a partition, P = $\{C_1, C_2\}$. Choose product s from $C_1$ and product t from $C_2$ such that

$$r_s = \max_{i \in C_1}(d_i) \text{ and } r_t = \min_{i \in C_2}(r_i) \quad (13)$$

Define $\delta = r_s - r_t$ and assume that $\delta \geq 0$. Let P' be a partition resulted from swapping product s and t. Since $d_i = d$ for all i, using equation (3), we have

$$E(SC_2 \mid P) = \frac{2}{D}\left(\frac{D_1}{R_1}\sum_{j \in A_1} t_j + \frac{D_2}{R_2}\sum_{j \in A_2} t_j\right) \quad (14)$$

$$E(SC_2 \mid P') = \frac{2}{D}\left(\frac{D_1}{R_1 - \delta}\sum_{j \in A_1 - A_r} t_j + \frac{D_2}{R_2 + \delta}\sum_{j \in A_2 + A_r} t_j\right) \quad (15)$$

where $A_r$ is the set of storage locations resulted from swapping two products and can be represented as $A_r = \{(R_1-\delta+1), (R_1-\delta+2), \cdots, R_1\}$ as shown in <Figure 1>. Note that $(R_1 - \delta) > 0$ since $R_1 \geq r_s > r_s - r_t = \delta$. Let $T_r$ be the expected SC travel time to $A_r$. Then, we have,

$$2\sum_{j \in A_r} t_j = \delta T_r \quad (16)$$

$$2\sum_{j \in A_k} t_j = R_k T_k \text{ for k=1, 2} \quad (17)$$

Using equation(16) and equation(17), we have,

$$v = E(SC_2 \mid P) - E(SC_2 \mid P')$$

$$= \frac{\delta\{D_1(R_2+\delta)(T_r - T_1) + D_2(R_1-\delta)(T_2-T_r)\}}{D(R_1-\delta)(R_2+\delta)} \quad (18)$$

Since $(T_r - T_1) \geq 0$, $(R_1 - \delta) \geq 0$, and $(T_2 - T_r) \geq 0$, v $\geq 0$. Hence if $r_s \geq r_t$, swapping two products s and t does not increase the expected SC travel time. In the similar manner, continuing to swap two products satisfying equation (13) and $\delta \geq 0$ results in a $r_i$-non-decreasing ordering or PAI-nonincreasing ordering eventually.

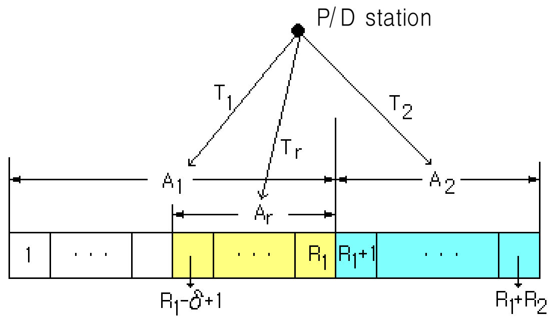


**Figure 1.** $A_r$ resulted from swapping two products.

As proved in the special cases above, $P^*$ is one of the partition based on a PAI-nonincreasing ordering. It follows that an optimal solution to the special cases can be represented as $(N_1^*)$ where $N_1^*$ denotes the first $N_1^*$ products of a PAI-nonincreasing ordering.

# 4. Counterexample

As discussed in Section 3, it seems likely that an optimal solution to PTN[2] is one of the partitions based on a PAI-nonincreasing ordering. However, this is not true since we have a counterexample as follows.

Counterexample : = $\{(r_i, d_i), i = 1, \cdots, 4\}$ = {(10, 9), (4, 0.7), (6, 1.0), (10, 1.0)}, $t_j$=1 for j=1,...,16 and 2 for j=17,...,30.

<Table 2> shows all possible partitions to the above counterexample. As shown in the table, $P^* = \{C_1^*, C_2^*\}$ = {{1, 3}, {2, 4}} with $E(SC_2^* \mid P^*)$=2.2906. It can be observed that $P^*$ is not based on PAI-nondecreasing ordering, {1, 2, 3, 4}.

# 5. A Heuristic Algorithm and Performance Bound

## 5.1 Some Basic Properties of PTN[2]

For convenience, define the activity index of class k $(CAI_k)$ as $\frac{D_k}{R_k}$. Consider the following property

**Table 2.** All possible 2-class-based storage layouts

| Partition | $D_1$ | $R_1$ | $CAI_1$ | $D_2$ | $R_2$ | $CAI_2$ | $E(SC_2)$ |
|---|---|---|---|---|---|---|---|
| +{{1},{2,3,4}} | 9.0 | 10 | 0.900 | 2.7 | 20 | 0.135 | 2.3231 |
| {{2},{1,3,4}} | 0.7 | 4 | 0.175 | 11.0 | 26 | 0.423 | - |
| {{3},{1,2,4}} | 1.0 | 6 | 0.167 | 10.7 | 24 | 0.446 | - |
| {{4},{1,2,3}} | 1.0 | 10 | 0.100 | 10.7 | 20 | 0.535 | - |
| +{{1,2},{3,4}} | 9.7 | 14 | 0.693 | 2.0 | 16 | 0.125 | 2.2991 |
| *{{1,3},{2,4}} | 10.0 | 16 | 0.625 | 1.7 | 14 | 0.121 | 2.2906* |
| +{{1,4},{2,3}} | 10.0 | 20 | 0.500 | 1.7 | 10 | 0.170 | 2.6325 |
| {{2,3},{1,4}} | 1.7 | 10 | 0.170 | 10.0 | 20 | 0.500 | - |
| {{2,4},{1,3}} | 1.7 | 14 | 0.121 | 10.0 | 16 | 0.625 | - |
| {{3,4},{1,2}} | 2.0 | 16 | 0.125 | 9.7 | 14 | 0.693 | - |
| +{{1,2,3},{4}} | 10.7 | 20 | 0.535 | 1.0 | 10 | 0.100 | 2.5368 |
| +{{1,2,4},{3}} | 10.7 | 24 | 0.446 | 1.0 | 6 | 0.167 | 2.7806 |
| +{{1,3,4},{2}} | 11.0 | 26 | 0.423 | 0.7 | 4 | 0.175 | 2.8429 |
| {{2,3,4},{1}} | 2.7 | 20 | 0.135 | 9.0 | 10 | 0.900 | - |

indicating that an optimal solution to PTN[2] is one of the partitions based on a CAI-nonincreasing ordering.

> Property 5. If $P^*=\{C_1^*, C_2^*\}$ is optimal to PTN[2], then $P^*$ satisfies $CAI_1^* \geq CAI_2^*$.

Proof: It suffices to show that if a partition, P= $\{C_1, C_2\}$, satisfies $CAI_1 < CAI_2$, then P is not optimal. Let $E(SC_K)$ and $E(SC_K^*)$ be the expected SC travel time and the expected minimum SC travel time given K classes respectively. Since $E(SC_2) \geq E(SC_1)$ if and only if $CAI_1 \geq CAI_2$, it follows that there exists $P^*$ such that $E(SC_2 \mid P) > E(SC_1) \geq E(SC_2 \mid P^*)$. Thus P is not optimal.

From Property 5, in order to obtain an optimal solution to PTN[2], it suffices to enumerate the partitions satisfying $CAI_1 \geq CAI_2$. However, it can be shown that the enumeration method is still intractable since the total number of partitions based on a CAI-nonincreasing ordering is not a polynomial function of n, i. e., $(2^{n-1}-1)$. Note that the number of feasible solutions to PTN[2] is $(2^n-2)$. Now consider the following property assuming that $CAI_1 \geq CAI_2$.

> Property 6. For $CAI_1 \geq CAI_2$,
> (i) If there exists product i in $C_2$ such that $PAI_i \geq CAI_1$, then assign product i to class 1 does not increase the expected SC travel time.
> (ii) If there exists product i in $C_1$ such that $PAI_i \geq$, $CAI_2$then assign product i to class 2 does not increase the expected SC travel time.

Proof: (i) Consider a partition P = $\{C_1, C_2\}$. Then $E(SC_2 \mid P)$ can be written as,

$$E(SC_2 \mid P)=\frac{D_1}{D}\left(\frac{1}{R_1}\sum_{j\in A_1} t_j\right)+\frac{D_2}{D}\left(\frac{1}{R_2}\sum_{j\in A_2} t_j\right) \quad (19)$$

Suppose that product i with $PAI_i=\frac{d}{r} \geq CAI_1$ has been assigned to $C_2$. Now, reallocate product i from $C_2$ to $C_1$. Let P' be the resulting partition. Let $A_r$ be the set of storage locations extended in $A_1$ by moving product i to class 1, i. e., $A_r=\{R_1+1, R_1+2, \cdots, R_1+r\}$. Then, the number of storage locations for class 1 and class 2 in P' will be $(R_1+r)$, $(R_2-r)$, and their corresponding retrieval rates will be $(D_1+d)$, $(D_2-d)$, respectively. Hence $E(SC_2 \mid P')$ can be expressed as,

$$E(SC_2 \mid P')=\frac{D_1+d}{D}\left(\frac{1}{R_1+r}\sum_{j\in A_1\cup A_r} t_j\right)+\frac{D_2-d}{D}\left(\frac{1}{R_2-r}\sum_{j\in A_2-A_r} t_j\right) \quad (20)$$

Let $T_k$ and $T_r$ be the expected SC travel time to zone k in P and $A_r$ respectively. Since $\sum_{j\in A_1\cup A_r} t_j = R_1T_1+rT_r$ and $\sum_{j\in A_2-A_r} t_j=R_2T_2-rT_r$, we have

$$\begin{aligned} v &= E(SC_2 \mid P)-E(SC_2 \mid P') \\ &= \frac{r}{(R_1+r)(R_2-r)D}R_1(R_2-r)(T_r-T_1) \\ &\quad (PAI_i-CAI_1)+R_2(R_1+r)(T_2-T_r) \\ &\quad (PAI_i-CAI_2) \end{aligned} \quad (21)$$

Since $R_2-r>0$, $T_r-T_1\geq 0$, $T_2-T_r\geq 0$, $PAI_i \geq CAI_1 \geq CAI_2$, it follows that $v \geq 0$.

(ii) Now, move product i with $PAI_i=\frac{d}{r} \leq CAI_2$ from $C_1$ to $C_2$. Let P'' be the resulting partition. In the similar manner above, we have,

$$\begin{aligned} v' &= E(SC_2 \mid P)-E(SC_2 \mid P'') \\ &= \frac{r}{(R_1-r)(R_2+r)D}R_1(R_2+r)(T'_r-T_1) \\ &\quad (CAI_1-PAI_i)+R_2(R_1-r)(T_2-T'_r) \\ &\quad (CAI_2-PAI_i) \end{aligned} \quad (22)$$

where $T'_r$ is the expected SC travel time to $A'_r=\{R_1-r+1, \cdots, R_1\}$. Since $R_1-r>0$, $T'_r-T_1\geq 0$, $T_2-T'_r\geq 0$, $PAI_i\leq CAI_2\leq CAI_1$, it follows that v' $\geq 0$.

From the above property, a property below can be derived, which can be used for checking whether a partition to PTN[2] is optimal or not.

> Property 7.
> (i) If $PAI_i \geq CAI_1^*$, then $i\in C_1^*$.
> (i) If $PAI_i \leq CAI_2^*$, then $i \in C_2^*$.

Proof: Trivial from Property 6.

It can be observed that Property 7 fails the optimality check if there is at least a product with $PAI_i$ such that $CAI_1^* > PAI_i > CAI_2^*$. However, the following property never fails.

Property 8. If $P^{*}=\{C_1^*, C_2^*\}$ is optimal to PTN[2], then

for $i \in C_1^*$,

$R_1^*(R_2^*+r_i)(T'_r-T_1^*)(CAI_1^*-PAI_i)+$
$R_2^*(R_1^*-r_i)(T_2^*-T'_r)(CAI_2^*-PAI_i)\le 0$,

for $i \in C_2^*$,

$R_1^*(R_2^*-r_i)(T_r-T_1^*)(PAI_i-CAI_1^*)+$
$R_2^*(R_1^*+r_i)(T_2^*-T_r)(PAI_i-CAI_2^*)\le 0$.

Proof: Trivial from both equation (21) and equation (22) in Property 6.

## 5.2 A Heuristic Algorithm and an Example

We state a heuristic algorithm, ALGPTN[2], which consists of three phases; initialization phase, local search phase, and optimality checking phase. In the initialization phase, we find a PAI-nonincreasing ordering, $O_{PAI}$, and find a starting solution, $\{C_{T1}, C_{T2}\}$ and $(N_{T1})$. In the local search phase, we generate a set of candidate solutions as follows. As proved in Property 6 and 7, product i with $PAI_i$ such that $CAI_1^* > PAI_i > CAI_2^*$ may or may not be assigned to $C_1^*$. Hence we find $N_L$ and $N_H$ such that $\frac{d_{N_L}}{r_{N_L}} < CAI_{T1}$ and $\frac{d_{N_H}}{r_{N_H}} > CAI_{T2}$ and the products which need to be swapped are products $N_L$-th through $N_H$-th in $O_{PAI}$. The number of swaps is determined as an integer constant $\delta$ such that $\delta = \min\{(N_{T1}-N_L+1), (N_H-N_{T1}+1)\}$. Next, for i = 1, ..., $\delta$, we swap $(N_L+i-1)$-th product in $C_{T1}$ with $(N_{T1}+i)$-th product in $C_{T2}$ and find our solution $\{C_1^o, C_2^o\}$ which may or may not give the minimum expected SC travel time. In the optimality checking phase, the optimality is checked using Property 8. Our algorithm can be summarized as below.

ALGPTN[2]:

(Initialization Phase)
Step 1. ESCMIN ← big value, $P_T \leftarrow \{C_{T1}, C_{T2}\} = \{\varnothing, \varnothing\}$
Step 2. Take products by PAI-nonincreasing order, $O_{PAI} = \{1, 2, \cdots, n\}$.
Step 3. For i=1 to (n-1), do
  Begin
    $P_T \leftarrow \{C_{T1}, C_{T2}\} = \{\{1,\cdots,i\},\{i+1,\cdots,n\}\}$
    Compute $E(SC_2 \mid P_T)$
    If( $E(SC_2 \mid P_T) <$ ESCMIN ) then
    Begin
      ESCMIN ← $E(SC \mid P_T)$
      $N_{T1} \leftarrow i$
    End
  End
  $P_T \leftarrow \{C_{T1}, C_{T2}\} = \{\{1,2,\cdots,N_{T1}\},\{N_{T1}+1, N_{T1}+2,\cdots,n\}\}$
  Compute $(CAI_{T1}, CAI_{T2})$.

(Local Search Phase)
Step 4. (Generate a set of candidate solutions, $C_X$ from $N_{T1}$.)
  Find the first $N_L$ and the first $N_H$ such that $\frac{d_{N_L}}{r_{N_L}} < CAI_{T1}$ and $\frac{d_{N_H}}{r_{N_H}} > CAI_{T2}$.
Step 5. (Swapping)
  $\delta \leftarrow \min((N_{T1}-N_L+1),(N_H-N_{T1}+1))$
  $P^o = \{C_1^o, C_2^o\} \leftarrow \{C_{T1}, C_{T2}\}$
  If $(\delta \ne 0)$ then
  Begin
    For i=1 to $\delta$, do
    Begin
      $C_1 \leftarrow C_{T1} - \{N_L+i-1\} + \{N_{T1}+i\}$
      $C_2 \leftarrow C_{T2} - \{N_{T1}+i\} + \{N_L+i-1\}$
      Compute $E(SC_2 \mid P)$
      If( $E(SC_2 \mid P) <$ ESCMIN ) then
      Begin
        ESCMIN ← $E(SC \mid P)$
        $P^o = \{C_1^o, C_2^o\} \leftarrow \{C_1, C_2\}$
      End
      $C_1, C_2 \leftarrow \varnothing$
    End
  End

(Optimality Check Phase)
Step 6. If Property 8 holds, then print "The solution, $\{C_1^o, C_2^o\}$ is optimal with $E(SC \mid P^o)$"
  otherwise, print "The solution, $\{C_1^o, C_2^o\}$ is near optimal with $E(SC \mid P^o)$"

Proposition 9. The time complexity of ALGPTL[2] is $O(n^2)$.

Proof : It can be checked that Step 1, 4, and 6 requires O(n), and Step 2 requires O(n log n), Step 3 and Step 5 requires $O(n^2)$. It follows that the time complexity of ALGPTN[2] is $O(n^2)$.

Example : $\{(r_i, d_i)\}$={(10, 9), (4, 0.7), (6, 1.0), (10, 1.0)}, $t_j$=1 for j=1, ..., 16 and 2 for j=17, ..., 30.

(Initialization Phase)
Step 1. ESCMIN $\leftarrow$ big value
Step 2. $O_{PAI}$={1, 2, 3, 4}
Step 3. $E(SC_2 \mid \{\{1\},\{2,3,4\}\}) \leftarrow 2.3231$
$E(SC_2 \mid \{\{1,2\},\{3,4\}\}) \leftarrow 2.2991$
$E(SC_2 \mid \{\{1,2,3\},\{4\}\}) \leftarrow 2.5368$
$N_{T1} \leftarrow 2$
$P_T \leftarrow \{C_{T1}, C_{T2}\} = \{\{1,2\},\{3,4\}\}$
$(CAI_{T1}, CAI_{T2}) \leftarrow (0.693, 0.125)$

(Local Search Phase)
Step 4. $N_L \leftarrow 2$ since $\frac{d_2}{r_2} = 0.175 < 0.693 = CAI_{T1}$
$N_H \leftarrow 3$ since $\frac{d_3}{r_3} = 0.167 < 0.125 = CAI_{T2}$
Step 5. (Swapping)
$\delta \leftarrow \min((N_{T1} - N_L + 1), (N_H - N_{T1}))$
= min(1, 1) =1
$C_1 \leftarrow C_{T1} - \{N_L + i - 1\} + \{N_{T1} + i\}$
= {1, 2} - {2} + {3} = {1, 3}
$C_2 \leftarrow C_{T2} - \{N_{T1} + i\} + \{N_L + i - 1\}$
= {3, 4} - {3} + {2} = {2, 4}
$E(SC_2 \mid \{\{1,3\},\{2,4\}\} \leftarrow 2.2906 <$ ESCMIN
= 2.2991
$P^o = \{C_1^o, C_2^o\} \leftarrow \{\{1, 3\}, \{2, 4\}\}$

(Optimality Check Phase)
Step 6. Since Property 8 holds, "The solution, {{1, 3}, {2, 4}} is optimal with 2.2906"

### 5.3 Performance Ratio Bound

Consider $\eta$, the ratio of $E(SC_2 \mid P^o)$ to $E(SC_2 \mid P^*)$ as shown in the following proposition.

Proposition 10.

$$\eta = \frac{E(SC_2 \mid P^o)}{E(SC_2 \mid P^*)} \le \frac{E(SC_1)}{E(SC_n^*)}$$

Proof : Since ALGPTN[2] enumerates a subset of the partitions based on a PAI-nonincreasing ordering, $P^o$ satisfies $CAI_1 \ge CAI_2$. Thus $E(SC_2 \mid P^o) \le E(SC_1)$. Since $E(SC_n^*) \le E(SC_2 \mid P^*)$, the proposition holds.

## 6. Conclusion

In this paper, we introduce a 2-class-based dedicated storage problem, PTN[2]. Since our strong conjecture is that PTN[2] is NP-hard, we provide three solvable cases including PTL[2] by relaxing PTN[2]. We prove that an optimal solution to the solvable cases is based on a PAI-nonincreasing ordering. Especially, we prove that an optimal solution to PTL[2] is based on a PAI-nonincreasing ordering and provide a greedy algorithm with O(n). Our first conjecture is that an optimal solution to PTN[2] is based on a PAI-nonincreasing ordering. However, we find with a counterexample that a solution based on the PAI index does not always give an optimal solution to PTN[2]. Nevertheless, using the PAI indexes still an effective approach to solving PTN[2] in the sense that it guarantees a better class-based storage layout than a randomized storage layout in terms of the expected SC travel time. Thus, an efficient heuristic algorithm, ALGPTN[2], for solving PTN[2] is constructed based on a PAI-nonincreasing ordering with performance ratio bound. In addition, some properties for PTN[2] are analyzed.

As discussed in this paper, our strong conjecture is that PTN[2] is NP-hard. This can be investigated further. Also, the performance ratio bound could be improved for the tighter bound or a worst-case bound in order to show that our ALGPTN[2] works well theoretically.

## Reference


Bozer, Y. A. and Cho, M. S. (1998), Throughput Performance of Automated Storage/Retrieval Systems under Stochastic Demand, *Working paper*, The University of Michigan, Ann Arbor, MI, 48109-2117.

Chang, D. T., Wen, U. P., and Lin, J. T. (1995), The Impact of Acceleration Deceleration on Travel Time Models for Automated Storage-Retrieval Systems, *IIE Transactions*, **27**(1), 108-111.

Cho, M. S. and Bozer, Y. A. (2001), Storage Capacity Estimation for Automated Storage/Retrieval Systems under Stochastic Demand, *Journal of Korean Institute of Industrial Engineers*, **27**(2), 169-175.

Francis, R. L. and White, J. A. (1974), Facility Layout and Location : an Analytical Approach, Prentice Hall.

Hausman, W. H., Schwartz, L. B., and Graves, S. C.(1976), Optimal Storage Assignment in Automatic Warehousing Systems, *Management Science*, Vol.22, No.6, pp629-638.

Lee, M. K. (1998), An Approach to Determining Storage Capacity of an Automated Storage/Retrieval System under Full Turnover Based Policy, *Journal of Korean Institute of Industrial Engineers*, **24**(4), 579-589.

Tompkins, J. A. and White, J. A. (1984), *Facilities Planning*, John Wiley and Sons Inc., NY., pp 335-338.

Yang, M. (1992), Optimization of Layout Design in an AS/RS for Maximizing its Throughput Rate, *Journal of the Korea Institute of Industrial Engineers*, Vol.18, No.2.