Journal of the Korean Institute of Industrial Engineers
Vol. 30, No. 3, pp. 190-196, September 2004.

Optimization of a 3-Class-based Dedicated Linear Storage System

Moonhee Yang† · Sun-uk Kim

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

3지역 / 지정위치 일차선형 저장시스템의 최적화

양문희 · 김선욱

단국대학교 공학부(산업공학전공)

In this paper, we address a layout design problem, PTL[3], for determining an optimal 3-class-based dedicated linear storage layout in a class of unit load storage systems. Our objective is to minimize the expected single command travel time.

We analyze PTL[3] to derive a fundamental property that an optimal solution to PTL[3] is one of the partitions based on the PAI(product activity index)-nonincreasing ordering. Using the property and partial enumeration, we construct an efficient exact algorithm with $O(n\lceil \log n \rceil)$ for solving PTL[3].

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. 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 and Bozer(2001), Lee(1998), Bozer and Cho(1998), Chang, Wen and Lin(1995), and Hausman, Schwartz and Graves(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. Yang(2003) suggested a deterministic 2-class-based dedicated storage problem and provide a heuristic algorithm with $O(n^2)$ as well as a greedy exact algorithm by restricting one-way travel time to a linear form.

In this paper, we define a 3-class-based dedicated linear storage problem, PTL[3], in a unit load system, and provide an efficient exact algorithm in addition to some basic properties. In Section 2, we describe PTL[3] in detail. In Section 3, we prove a fundamental property that an optimal solution to PTL[3] is one of the partitions based on the PAI(product activity index)-nonincreasing ordering, and provide useful properties required for constructing our algorithm. In Section 4, based on the properties and partial enumeration, we construct an efficient exact algorithm with $O(n\lceil \log n \rceil)$ for solving PTL[3] and we give an example and additional comments.

† Corresponding author : Professor Moonhee Yang, Department of Industrial Engineering, Dankook University, Cheonan, 330-714,
Fax +82-41-550-3570; E-mail myfriend@dankook.ac.kr
Received November 2003; revision received April 2004; accepted May 2004.

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. Description of a 3-Class-Based Linear 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 3-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 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, 3. Given the t_j-nondecreasing ordering, we assign the first $|A_1|$ storage locations to A_1, and assign the next $|A_2|$ storage locations to A_2, and the remaining storage locations to A_3 where $|X|$ denotes the cardinality of set X. It follows that $A_1 = \{1, 2, \cdots, |A_1|\}$, $A_2 = \{|A_1|+1, |A_1|+2, \cdots, |A_1|+|A_2|\}$, and $A_3 = \{|A_1|+|A_2|+1, |A_1|+|A_2|+2, \cdots, R\}$.

An arriving replenishment lot of a product i, the size of which is r_i in unit load, contains a single product and is assigned randomly to open storage locations in one of three 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

$$R_k = |A_k| = \sum_{i \in C_k} r_i \qquad (1)$$

It can be observed that the number of storage locations for a class is not the maximum aggregate

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
$CAI(N_1, N_2)$	$\sum_{i=N_1}^{N_2} d_i / \sum_{i=N_1}^{N_2} r_i$ given a PAI-nonincreasing ordering
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
K	number of classes or zones used in a unit load system
n	number of products
N_k	number of products assigned to classes 1 through k
O	PAI-nonincreasing ordering
P(K)	$\{C_1, C_2, \cdots, C_K\}$, a partition given K classes
PAI_i or $PAI(i)$	$\frac{d_i}{r_i}$, 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	$|A_k|$, number of storage locations required for zone k
t_j	one-way travel time to storage location j, j=1, ..., R
T_k	expected SC travel time from an i/o point to zone k

inventory position for a class but the sum of space requirement for products assigned to the class. In fact, the implicit "constant-space assumption" is made since the problem for minimizing the maximum aggregate inventory position is known to be NP-hard [Hall (1987)] and it can make PTL[3] more complicated.

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 $|C_k| \geq 1$.

Our objective is to minimize $E(SC_3)$, the expected single command(SC) travel time as follows. The expected SC travel time to zone k, T_k, can be expressed as

$$T_k = \frac{2}{|A_k|} \sum_{j \in A_k} t_j \qquad (2)$$

Since the probability of visiting zone k is $\frac{D_k}{D}$, $E(SC_3)$ can be expressed as

$$E(SC_3) = \sum_{k=1}^{3} \frac{D_k}{D} T_k \qquad (3)$$

$$= \frac{2}{D} \sum_{k=1}^{3} \frac{D_k}{|A_k|} \sum_{j \in A_k} t_j \qquad (4)$$

where $D = \sum_{k=1}^{3} D_k$. By replacing t_j with pj+q for constants p and q, Eq.(4) can be further reduced as

$$E(SC_3) = \frac{p}{D}\{D + DR + (R_1D_2 - D_1R_2) + (R_2D_3 - D_2R_3) + (R_1D_3 - D_1R_3)\} + 2q \qquad (5)$$

where $R = \sum_{k=1}^{3} R_k$.

It can be observed that $E(SC_3)$ does not depend on t_j but on (r_i, d_i). Since D, R, p and q are constant and each class must contain at least one product, PTL[3] can be stated as

PTL[3] : Given n products with $\{(r_i, d_i), i = 1, 2, \cdots, n\}$, find an optimal partition, $P^*(3) = \{C_1^*, C_2^*, C_3^*\}$ such that we

$$\text{Minimize } Z = (R_1D_2 - D_1R_2) + (R_2D_3 - D_2R_3) + (R_1D_3 - D_1R_3)$$

$$\text{subject to } |C_k| \geq 1 \text{ for k=1, 2, 3}$$

$$D_k = \sum_{i \in C_k} d_i \text{ for k=1, 2, 3}$$

$$R_k = \sum_{i \in C_k} r_i \text{ for k=1, 2, 3}$$

3. Basic Properties of PTL[3]

3.1 A Necessary Condition

In order to facilitate our analysis, define PTL[K] to be the K-class-based dedicated linear storage problem and define $P^*(K)$ or $\{C_1^*, C_2^*, \cdots, C_K^*\}$ to be an optimal solution to PTL[K]. Let $CAI(N_1, N_2)$ be $\sum_{i=N_1}^{N_2} d_i / \sum_{i=N_1}^{N_2} r_i$.

Yang(2003) analyzed PTL[2] and derived a necessary and sufficient condition to PTL[2]. For convenience, we state his result in Proposition 1 as follows.

> Proposition 1. $P^*(2)$ is optimal to PTL[2] if and only if $P^*(2)$ satisfies
> $PAI(i_1) \geq CAI(1,n) > PAI(i_2)$
> where $i_k \in C_k^*$ for k=1, 2.

Proposition 1 implies that if we take products by PAI-nonincreasing order and if we assign the first N_1 products, each PAI value of which is greater than or equal to $CAI(1,n)$, to C_1^* and the remaining products to C_2^*, then $\{C_1^*, C_2^*\}$ is optimal. In other words, $P^*(2)$ is one of the partitions based on a PAI-nonincreasing ordering.

The natural question will be whether $P^*(K)$ is one of the partitions based on a PAI-nonincreasing ordering or not. In what follows, we prove a necessary condition that $P^*(3)$ is also one of the partitions based on a PAI-nonincreasing ordering. From now on, we assume that a PAI-nonincreasing ordering is given as $O = (1,2,\cdots,n)$. Given O, define any candidate solution to PTL[3] as $X(3) = (N_1, N_2)$ where N_k denotes the number of products assigned to classes 1 through k for k = 1, 2, 3.

> Proposition 2. (i) If $P^*(3)$ is optimal to PTL[3], then $P^*(3)$ satisfies
> $PAI(i_1) \geq CAI(1, N_2^*) > PAI(i_2)$
> $\geq CAI(N_1^* + 1, n) > PAI(i_3)$
> where $i_k \in C_k^*$ for k=1, 2, 3.
> (ii) $P^*(3)$ is one of the partitions based on a PAI-nonincreasing ordering.

Proof : Applying Proposition 1 to classes C_1^* and C_2^*, for $i_1 \in C_1^*$, $i_2 \in C_2^*$, we have

$$PAI(i_1) \geq CAI(1, N_2^*) > PAI(i_2) \quad (6)$$

Similarly, applying Proposition 1 to classes C_2^* and C_3^*, for $i_2 \in C_2^*$, $i_3 \in C_3^*$, we have

$$PAI(i_2) \geq CAI(N_1^* + 1, n) > PAI(i_3) \quad (7)$$

If both Eq.(6) and Eq.(7) do not hold, $E(SC_3)$ can be further reduced by swapping two products which violate either Eq.(6) or Eq.(7). This is a contradiction to that $P^*(3)$ is optimal. Hence both Eq.(6) and Eq.(7) hold. By rearranging products in each class by PAI-nonincreasing order, finally $P^*(3)$ will be one of the partitions based on a PAI-nonincreasing ordering. This completes the proof.

Proposition 2 implies that $P^*(K)$ is also one of the partitions based on a PAI-nonincreasing ordering. Hence, $P^*(K)$ can be obtained by enumerating all the possible partitions based on a PAI-nonincreasing ordering.

Now, consider PTL[3]. In order to find a $P^*(3)$, it is enough to enumerate (N_1, N_2). The total number of candidate solutions will be $\frac{(n-1)(n-2)}{2}$ since the number of candidate solutions given N_1 is $(n - N_1 - 1)$ where $1 \leq N_1 \leq (n-2)$. That is, the total enumeration requires $O(n^2)$. However, the total enumeration can be further reduced by introducing "a conditional local optimal solution" as follows.

3.2 Conditional Local Optimal Solution

Suppose that the first N_1 products of a PAI-nonincreasing ordering have been assigned to class 1. Then, N_2 can be any value such that $N_1 + 1 \leq N_2 \leq (n-1)$. The next question will be "What value of N_2 gives the minimum $E(SC_3)$ given N_1?" This question corresponds to finding a local optimal solution from a set of solutions;

$$\{(N_1, N_1+1), (N_1, N_1+2), \cdots, (N_1, n-1)\}$$

Clearly the minimization of $E(SC_3)$ given N_1 is equivalent to solving a 2-class-based dedicated linear storage problem with the last $(n - N_1)$ products of the PAI-nonincreasing ordering. Let $(N_1, N_2^c(N_1))$ be the local optimal solution given N_1. Then from Proposition 1, $N_2^c(N_1)$ can be determined such that

$$PAI(N_2^c(N_1)) \geq CAI(N_1 + 1, n) > PAI(N_2^c(N_1) + 1) \quad (8)$$

Similarly, the minimization of $E(SC_3)$ given the last $(n - N_2)$ products of a PAI-nonincreasing ordering assigned to class 3 is equivalent to solving a 2-class-based dedicated linear storage problem with the first N_2 products of the PAI-nonincreasing ordering. Let $(N_1^c(N_2), N_2)$ be the local optimal solution given N_2. Then, from Proposition 1, $N_1^c(N_2)$ can be determined such that

$$PAI(N_1^c(N_2)) \geq CAI(1, N_2) > PAI(N_1^c(N_2) + 1) \quad (9)$$

Hence, in order to find a $P^*(3)$, it suffices to enumerate all the local optimal solutions in either set E_1 or set E_2 as follows:

$$E_1 = \{(N_1, N_2^c(N_1)),\ N_1 = 1, \cdots, (n-2)\} \quad (10)$$

$$E_2 = \{(N_1^c(N_2), N_2),\ N_2 = 2, \cdots, (n-1)\} \quad (11)$$

Now, we prove that $N_2^c(N_1^*)$ is N_2^*, and that $N_1^c(N_2^*)$ is N_1^*.

> Property 1. Given an O,
> (i) $N_2^c(N_1^*) = N_2^*$
> (ii) $N_1^c(N_2^*) = N_1^*$

Proof : (i) Since $PAI(N_2^c(N_1^*)) \geq CAI(N_1^* + 1, n)$ (From Eq.(8))
$> PAI(N_2^* + 1)$ (From Proposition 2, (i)),
we have

$$N_2^c(N_1^*) \leq N_2^* \quad (12)$$

Similarly, since $PAI(N_2^*) \geq CAI(N_1^* + 1, n)$ (From Proposition 2, (i)),
$> PAI(N_2^c(N_1^*) + 1)$ (From Eq.(8)
we have

$$N_2^* \leq N_2^c(N_1^*) \quad (13)$$

Therefore, from Eq.(12) and Eq.(13), $N_2^c(N_1^*) = N_2^*$. In the similar manner, (ii) can be proved. This completes the proof.

3.3 Lower and Upper Bounds of N_1^*

If a lower and an upper bounds of N_1^* are available, we can further reduce the enumeration. Before we

derive a lower and an upper bounds on N_1^*, we need the following property.

> Property 2. Given an O,
> (i) $N_2^c(N_1)$ is a nondecreasing function of N_1.
> (ii) $N_1^c(N_2)$ is a nondecreasing function of N_2.

Proof : (i) It is enough to prove that $N_2^c(N_1) \le N_2^c(N_1+1)$. Since

$$PAI(N_2^c(N_1)) \ge CAI(N_1+1, n) \text{ (From Eq.(8))}$$
$$\ge CAI(N_1+2, n)$$
$$> PAI(N_2^c(N_1+1)+1) \text{(From Eq.(8))},$$

we have $PAI(N_2^c(N_1)) > PAI(N_2^c(N_1+1)+1)$, i.e., $N_2^c(N_1) \le N_2^c(N_1+1)$. In the similar manner, (ii) can be proved. This completes the proof.

Now, consider the following property for a lower and an upper bounds of N_1^*. Let L_k and U_k be a lower and an upper bounds of N_k^* for k=1, 2 respectively.

> Property 3. Given an O, L_1 and U_1 can be obtained by solving the equations below:
> (i) $PAI(L_1) \ge CAI(1, N_2^c(1)) > PAI(L_1+1)$
> (ii) $U_1 = N_1^c(n-1)$

Proof: (i) Since a possible minimum value of L_1 is 1, $N_2^c(1) \le N_2^c(L_1) \le N_2^*$. That is, a possible minimum value of N_2^* is $N_2^c(1)$. Using Proposition 2, (i), we can obtain L_1 such that

$$PAI(L_1) \ge CAI(1, N_2^c(1)) > PAI(L_1+1) \quad (14)$$

(ii) Since a possible maximum value of U_2 is (n-1), $N_1^c(n-1)$ can be an upper bound of N_1^*. In a formal way, since $N_2^* \le U_2$ and $N_1^c(N_2)$ is a nondecreasing function of N_2, we have

$$N_1^c(N_2^*) \le N_1^c(U_2) \quad (15)$$

Since $N_1^* = N_1^c(N_2^*)$ from Property 1, (ii), and a possible maximum value of U_2 is (n-1), it follows that

$$N_1^c(N_2^*) = N_1^* \le N_1^c(U_2) \le N_1^c(n-1) \quad (16)$$

Thus, U_1 can be obtained from Eq.(16) as

$$U_1 = N_1^c(n-1) \quad (17)$$

This completes the proof.

4. An Exact Algorithm and an Example

4.1 Exact Algorithm

Based on Proposition 2 and properties proved above, an efficient exact algorithm, ALGPTL[3], which solves PTL[3], can be constructed as follows:

ALGPTL[3]
Step 1. (Initialization Phase)
  Take products by PAI-nonincreasing order.
  $E(SC_3^*) \leftarrow$ big value
  Compute (L_1, U_1) using Property 3.
Step 2. (Optimization Phase)
  For $N_1 = L_1$ to U_1, do
  Begin
    Find $N_2^c(N_1)$ such that $PAI(N_2^c(N_1)) \ge CAI$
      $CAI(N_1+1, n) > PAI((N_2^c(N_1)+1)$
    Compute $E(SC_3)$ using $(N_1, N_2^c(N_1))$
    If $E(SC_3) < E(SC_3^*)$, then $E(SC_3^*) \leftarrow$
      $E(SC_3)$ and $(N_1^*, N_2^*) \leftarrow (N_1, N_2^c(N_1))$
  End

> Proposition 3. ALGPTL[3] solves PTL[3] in $O(n \lceil \log n \rceil)$.

Proof: Since ALGPTL[3] enumerates all the local optimal solutions given N_1 such that $L_1 \le N_1 \le U_1$, ALGPTL[3] solves PTL[3]. Now, Step 1 requires $O(n \lceil \log n \rceil)$, the maximum number of iterations of Step 2 is O(n) and finding $N_2^c(N_1)$ requires $O(\lceil \log n \rceil)$ since a PAI-nonincreasing ordering is given. It follows that Step 2 requires $O(n \lceil \log n \rceil)$. Therefore the time complexity of ALGPTL[3] is $O(n \lceil \log n \rceil)$. This completes the proof.

4.2 An Example and Additional Comments

Suppose that the input data are given as shown in <Table 2> and $t_j = j$ for j=1, ···, 161. Given N_1=1, $E(SC_3)$ changes depending on the value of N_2 as shown in <Table 3>. It can be clearly observed in <Figure 2> that $E(SC_3)$ decreases until N_2= 6 and increases after N_2= 6, and that $E(SC_3)$ given N_1= 1 is minimized when N_2= 6, i.e., $N_2^c(1)$= 6. In the similar manner, changing the value of N_1 from 1 to 8,

we can obtain the corresponding value, $N_2^c(N_1)$, to N_1 as shown in the second column of <Table 4>. Note that $N_2^c(N_1)$ is a nondecreasing function of N_1 which was proved in Property 2. From the table, we have $P^*(3) = (N_1^*, N_2^*) = (3, 7)$.

As stated in ALGPTL[3], we take products by PAI-nonincreasing order as shown in <Table 2>, and assign any big value to $E(SC_3^*)$. Using Property 3, (ii), we have $U_1 = 3$ because $N_1^c(9) = 3$. Note that $PAI_6 = 0.2 > CAI(2,10) = 0.1795 > PAI_7 = 0.1670$ and that $PAI_3 = 0.25 > CAI(1, 9) = 0.2252 > PAI_4 = 0.2$. Since $N_2^c(1) = 6$ and $PAI_3 = 0.25 > CAI(1,6) = 0.2471 > PAI_4 = 0.2$, we have $L_1 = 3$. Note that $N_1^* = 3$ in this example since $L_1 = U_1$.

Since $PAI_7 = 0.1670 > CAI(4, 10) = 0.1570 > PAI_8 = 0.1430$, $N_2^* = N_2^c(3) = 7$. Thus $X^*(3) = (3, 7)$, i.e., we assign products 1, 2 and 3 to class 1 and products 4, 5, 6 and 7 to class 2 and the remaining products to class 3, which gives $E(SC_3) = 131.6774$.

Table 3. $E(SC_3)$ depending on N_2 given $N_1 = 1$

N_2	$E(SC_3)$	N_2	$E(SC_3)$
2	144.8387	6*	133.0968*
3	137.7419	7	133.8710
4	136.7097	8	135.1613
5	135.1613	9	136.4516

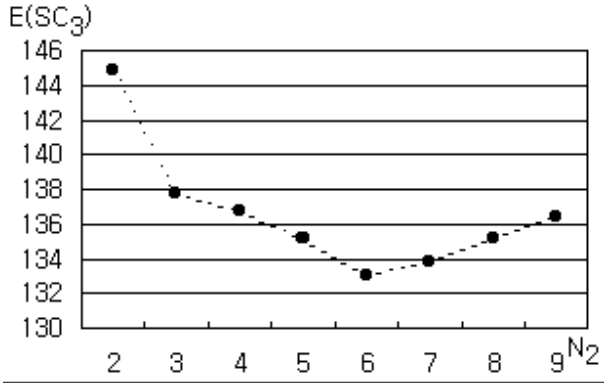

Figure 1. Local optimal solution given $N_1 = 1$.

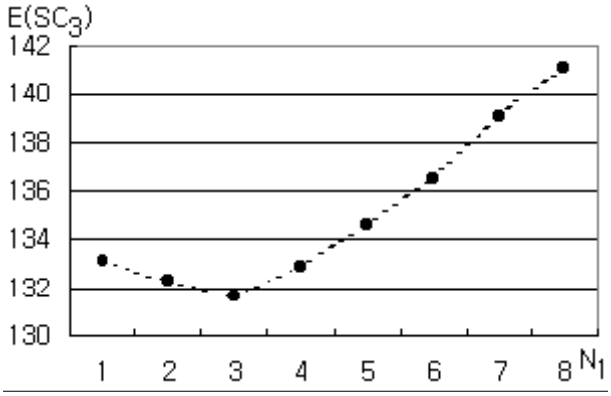

Figure 2. $E(SC_3)$ given $N_1 = 1, 2, ..., 8$ from the example.

Table 2. Input Data $\{(r_i, d_i)\}$

Product	r_i	d_i	PAI_i	Product	r_i	d_i	PAI_i
1	5	3	0.6000	6	20	4	0.2000
2	15	4	0.2670	7	12	2	0.1670
3	20	5	0.2500	8	7	1	0.1430
4	10	2	0.2000	9	7	1	0.1430
5	15	3	0.2000	10	50	6	0.1200

Table 4. Local optimal solutions given $N_1 = 1, 2, ..., 8$

N_1	$N_2^c(N_1)$	D_1	D_2	D_3	R_1	R_2	R_3	$E(SC_3)$
1	6	3	18	10	5	80	76	133.0968
2	6	7	14	10	20	65	76	132.2903
3*	7*	12	9	10	40	45	76	131.6774*
4	7	14	9	8	50	47	64	132.8387
5	7	17	6	8	65	32	64	134.5807
6	9	21	4	6	85	26	50	136.5161
7	9	23	2	6	97	14	50	139.0323
8	9	24	1	6	104	7	50	141.0968

5. Conclusions

In this paper, we introduce a 3-class-based dedicated linear storage problem, PTL[3], for determining an optimal 3-class-based dedicated linear storage layout in a class of unit load storage systems.

We analyze PTL[3] to derive a fundamental property that an optimal solution to PTL[3] is one of the partitions based on the PAI-nonincreasing ordering. Using the property and partial enumeration, we construct an efficient exact algorithm, ALGPTL[3], with $O(n \lceil \log n \rceil)$ for solving PTL[3]. Our algorithm could be utilized to construct a heuristic algorithm for solving a 3-class-based dedicated storage problem, which does not assume the linearity of the one-way travel time.

Our strong conjectures are that Proposition 2 could be a sufficient condition and that there might exist a greedy algorithm for PTL[3] like that for PTL[2]. These conjectures can be further investigated.

References

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, **48**, 109-2117.

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.

Hall, N. G., (1987) "A Comparison of Inventory Replenishment Heuristics for Minimizing Maximum Storage", Ohio state University, Working paper.

Hausman, W. H., Schwartz, L. B., and Graves, S. C. (1976), Optimal Storage Assignment in Automatic Warehousing Systems, *Management Science*, **22**(6), 629-638.

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., 335-338.

Yang, M. (2003), Analysis and Optimization of a 2-class-based Dedicated Storage System, *Journal of the Korea Institute of Industrial Engineers*, **29**(3), 222-229.