IE Interfaces
Vol. 22, No. 2, pp. 116-125, June 2009.

조립블록 저장위치 할당문제에 대한 재고찰

박창규† · 서준용

울산대학교 경영대학

On the Assembly Block Storage Location Assignment Problem

Changkyu Park · Junyong Seo

College of Business Administration, University of Ulsan

We revisit the assembly block storage location assignment problem (ABSLAP) at a shipyard, in order to compensate for the deficiency in performance verification of the heuristic ABSLAP algorithm developed by the previous study. In this paper, we formulate a mathematical programming model of the ABSLAP, refine elaborately the heuristic ABSLAP algorithm, and show the performance of the developed mathematical programming model and the revised heuristic ABSLAP algorithm. In addition, we explain simulation experiments conducted using the revised heuristic ABSLAP algorithm to investigate the influences of block stockyard layouts and production schedule instability on the block stockyard operations.

Keyword: block stockyard, storage location assignment problem, shipbuilding, heuristic algorithm, mathematical programming model

1. 서 론

본 논문은 Park and Seo(2006)가 처음으로 소개한 조선소에서의 조립블록 저장위치 할당문제(ABSLAP, assembly block storage location assignment problem)에 대해 재조명해 보고자 한다. Park and Seo(2006)는 현재 세계적으로 가장 큰 조선사인 현대중공업에서 수행한 프로젝트에 기초하여 조립블록 적치장 운영의 어려움을 지적하고, ABSLAP를 정형화함과 동시에 발견적 ABSLAP 알고리즘을 제시하였다. 그리고 다양한 상황에 대해 수행한 모의실험을 통하여 발견적 ABSLAP 알고리즘의 유용성을 입증하였다. 그렇지만 조금 아쉬운 점으로 남아 있었던 것은 발견적 ABSLAP 알고리즘을 수리모형 등과 같은 최적화 기법과 비교하여 성능을 검증해 보았으면 하는 것이었다. 따라서 본 논문은 우선 ABSLAP에 대한 수리모형을 세우고, 또한 기존의 발견적 ABSLAP 알고리즘을 보다 정교하게 가다듬은 다음(저장위치 결정 하위-알고리즘 중 단계 1 : 접근 가능한 후보 위치 탐색), 수립한 수리모형과 교정한 발견적 ABSLAP 알고리즘의 성능을 보여 주고자 한다.

조선소에서의 ABSLAP는 항만 컨테이너 터미널 운영을 다루는 연구 분야에서 소개한 컨테이너 저장 공간 및 위치 할당문제와 유사하다(예를 들어, Bazzazi *et al*., 2008; Kozan and Preston, 2006; Preston and Kozan, 2001; Zhang *et al*., 2003). 조선소에서의 ABSLAP와 항만 컨테이너 터미널에서의 저장과 관련된 문제는 기본적으로 같은 서비스, 즉 반입 및 반출되는 대상물(예를 들어, 컨테이너와 조립블록)에게 일시적인 저장 장소를 제공하는 서비스를 다룬다. 그러나 조선소에서의 ABSLAP와 항만 컨테이너 터미널에서의 저장과 관련된 문제는 운반 작업에서 큰 차이를 보인다. 다시 말하면, 컨테이너는 항만 크레인에 의해 이동될 수 있기 때문에 항만 컨테이너 터미널에서의 저장과 관련된 문제는 반입 및 반출되는 컨테이너를 3차원적으로(즉, X, Y, Z 방향) 이동시킬 수 있다. 그러나 조선소에서의 ABSLAP는 블록 적치장 운영에 크레인 사용이 허용되지 않기 때문에 반입 및 반출되는 조립블록을 오직 X 및 Y 방향의 2차원적으로만 이동 가능하다(조선소의 블록 적치장에서 크레인이 사용

이 논문은 2008년 울산대학교의 연구비에 의해 연구되었음.
†연락저자 : 박창규 교수, 680-749 울산광역시 남구 대학로 102 울산대학교 경영대학, Fax : 052-247-7619, E-mail : ckparkuou@ulsan.ac.kr
투고일(2008년 12월 19일), 심사일(1차 : 2009년 01월 14일, 2차 : 2009년 02월 09일), 게재확정일(2009년 02월 11일).

되지 않는 이유는 (a) 블록의 크기와 무게가 거대하고 무겁다. 일반적으로 15m×15m×5m 정도이고, 무게는 100에서 300톤가량이다. 최근 일반적인 블록의 크기와 무게를 초과하는 초대형 블록이 종종 발생한다; (b) 블록의 형태가 비정형화되어 있어서 여러 개의 블록을 포개어 쌓는 것이 불가능하다).

조선소에서의 ABSLAP에 대해 간단히 설명하면 다음과 같다. 먼저, <Figure 1>은 m×n (즉, 깊이×폭) 셀 (cell)로 구성된 블록 적치장의 개략적인 도식을 보여 준다. 각 셀은 한 개의 블록만을 저장할 수 있다. 즉, 한 셀에서 한 블록 위에 다른 블록을 쌓아 놓는 방식으로 여러 개를 동시에 저장하는 것은 허용되지 않는다. 따라서 블록 적치장의 최대 저장 용량은 m×n 으로 계산된다. 블록 적치장은 직사각형 모양의 체스판 (chessboard)과 비슷하다.

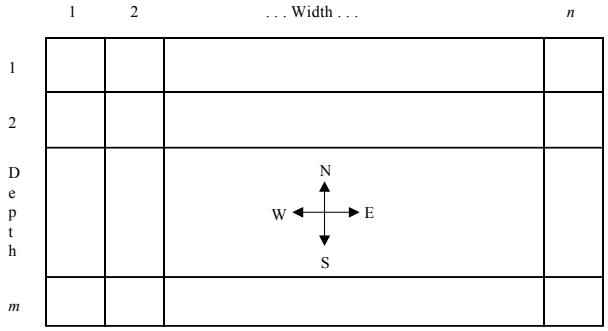


Figure 1. The schematic drawing of block stockyard

이와 같은 블록 적치장에서 블록 운반 작업을 효율적으로 수행하는 데 가장 큰 걸림돌은 블록들을 단지 X 및 Y 방향으로만 이동시킬 수 있고 Z 방향으로는 이동시킬 수 없다는 제약이다. 따라서 블록 적치장에 블록을 반입 및 반출시킬 때, 어떤 블록이 반입 및 반출되는 블록의 이동 경로 상에 존재한다면, 반입 및 반출되는 블록에게 자유로운 이동 경로를 제공하기 위해 방해되는 블록을 먼저 다른 빈 셀로 옮겨야 한다. 이러한 방해 블록의 운반 작업은 가급적 발생하지 않도록 피해야할 비생산적인 활동이다.

실제 상황에서 블록 적치장은 시간에 구애 받지 않고 연속적으로 운영된다. 그러나 본 논문은 연구 목적을 위해 연속적인 시간을 이산적인 단위 기간(예를 들어, 하루)으로 나눈다. 각 단위 기간별로 블록 적치장은 만기가 된 블록을 반출시키고 반입되는 블록을 저장한다. 반입되는 블록은 블록 적치장에 저장될 때 반출될 시기가 미리 계획되어 진다. 이러한 상황에서 ABSLAP는 방해 블록의 수가 최소로 발생하도록 만기된 블록을 블록 적치장으로부터 반출하고, 방해 블록 및 반입 블록에게 블록 적치장의 빈 셀을 할당하는 문제로 정의된다.

ABSLAP는 일반화된 할당문제 형태로 모형화할 수 있지만 이는 NP-hard 문제 범주에 속한다(Bazzazi *et al*., 2008; Preston and Kozan, 2002). 이 문제에 대한 계산적인 복잡성은 일정계획의 대상이 되는 블록의 수에 따라 기하급수적으로 증가한다. 따라서 이러한 문제에 대한 해를 지금까지 알려진 최적화 기법(예를 들어, Branch and Bound, Tree Searches 등)으로 합리적인 시간 내에 찾기란 어렵다. 이와 같은 상황은 큰 규모의 현실적인 문제를 풀기 위해 발견적 기법을 사용하는 것이 현명한 선택임을 의미한다. 본 논문은 너무 무리한 계산적 부담 없이 수리모형으로도 풀 수 있는 적절한 크기의 문제를 이용하여 발견적 ABSLAP 알고리즘의 성능을 검증하기 위해 ABSLAP의 수리모형을 비교 기준으로 활용한다.

앞으로 본 논문의 구성은 다음과 같이 되어 있다. 먼저 제 2장은 ABSLAP에 대해 수립한 수리모형에 관하여 설명하고, 제 3장은 교정한 발견적 ABSLAP 알고리즘을 설명한다. 다음으로 제 4장은 컴퓨터 실험 결과를 설명하고, 결론은 마지막 장에서 제시한다.

2. ABSLAP의 수리모형

이 장에서는 다음의 가정 사항들을 기반으로 조립블록 저장위치 할당문제 (ABSLAP)를 수리적으로 모형화 한다. ABSLAP의 주목표는 방해 블록의 수가 최소로 발생하도록 블록 적치장에 블록을 반입 및 반출시키는 것이다.

2.1 가정 사항

(1) 블록 적치장에서 블록은 일직선 방향으로만 움직일 수 있다. 즉, 블록은 지그재그 형태로 움직일 수 없다.
(2) 각 단위 기간에 반출할 블록을 먼저 블록 적치장으로부터 출하하고 난 후, 반입 블록을 블록 적치장에 저장한다.
(3) 어떤 블록을 반입 및 반출할 때 발생하는 방해 블록은 먼저 블록 적치장으로부터 빼낸다. 그리고 반입 및 반출 작업이 완료된 후, 블록 적치장으로부터 임시로 빼낸 방해 블록을 원위치 시킨다.
(4) 반입되는 블록 및 블록 적치장에 머물게 될 기간에 대한 모든 정보는 연구 대상 기간 동안 변하지 않는다.

2.2 첨자 및 입력 모수

i 블록, $i = 1, 2, \cdots, I$
j 저장 위치, $j = 1, 2, \cdots, J$ ($J = m \times n$)
k 블록 이동 경로 상에 있는 저장 위치
l 블록 이동 방향
t 시간(단위 기간), $t = 1, 2, \cdots, T$
IB_t 시간 t 에 반입될 블록의 집합
OB_t 시간 t 에 반출될 블록의 집합
MD_j 저장 위치 j 에서 블록 이동 방향의 집합
MP_{lj} 저장 위치 j 에서부터 블록 이동 방향 l 상에 있는 저장 위치의 집합

S_i 블록 i의 반입 시간
F_i 블록 i의 반출 시간
m 블록 적치장의 깊이(행의 수)
n 블록 적치장의 폭(열의 수)

2.3 의사 결정 변수

X_{ijt} 1, 블록 i가 시간 t에 저장 위치 j에 할당될 경우; 0, 그 외의 경우
Y_{jt} 1, 시간 t에 저장 위치 j에 블록이 존재할 경우; 0, 그 외의 경우

2.4 수리모형

$$Minimize \sum_{t=1}^{T}\sum_{i\in OB_t}\sum_{j=1}^{J} \min_{l\in MD_j}\left\{\sum_{k\in MP_{lj}} X_{ijt}Y_{kt}\right\} + \sum_{t=1}^{T}\sum_{i\in IB_t}\sum_{j=1}^{J} \min_{l\in MD_j}\left\{\sum_{k\in MP_{lj}} X_{ijt}Y_{kt}\right\} \quad (1)$$

Subject to

$$\sum_{j=1}^{J} X_{ijt} = 1 \quad \text{for } i\in IB_t, \forall t \quad (2)$$

$$X_{ijt}X_{i'jt} = 0 \quad \text{for } i < i', i \text{ and } i' \in IB_t, \forall j, \forall t \quad (3)$$

$$X_{ijt}Y_{jt} = 0 \quad \text{for } i\in IB_t, \forall j, \forall t \quad (4)$$

$$Y_{jt'} = \max\{Y_{jt'}, X_{ijt}\} \quad \text{for } t' = S_i+1, \cdots, F_i-1, i\in IB_t, \forall j, \forall t \quad (5)$$

$$X_{ijF_i} = X_{ijt} \quad \text{for } i\in IB_t, \forall j, \forall t \quad (6)$$

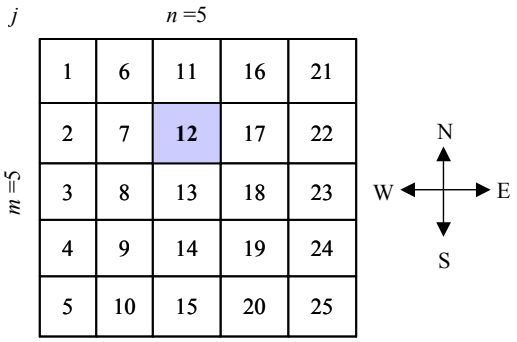
모든 의사 결정 변수는 0 또는 1

(1) 목적함수

ABSLAP의 목적함수는 방해 블록의 수를 최소화하는 것이다. 블록 적치장 운영에서 방해 블록이 발생할 수 있는 경우는 두 가지로 블록 적치장으로부터 반출할 블록을 출하할 때와 블록 적치장에 반입할 블록을 저장할 때이다. 식 (1)은 각 경우에 발생할 방해 블록의 수를 계산하는 방법을 보여준다.

$$Minimize \sum_{t=1}^{T}\sum_{i\in OB_t}\sum_{j=1}^{J} \min_{l\in MD_j}\left\{\sum_{k\in MP_{lj}} X_{ijt}Y_{kt}\right\} + \sum_{t=1}^{T}\sum_{i\in IB_t}\sum_{j=1}^{J} \min_{l\in MD_j}\left\{\sum_{k\in MP_{lj}} X_{ijt}Y_{kt}\right\} \quad (1)$$

<Figure 2>는 크기가 5×5인 블록 적치장에서 방해 블록의 수를 계산하기 위해 어떤 셀들을 점검해야 하는 지를 설명하는 예제를 보여주고 있다. 예를 들어, 블록의 반출 및 반입이 오직 블록 적치장의 한쪽 면(예를 들어, 남쪽 방향)으로만 제한되고 반출 또는 반입될 블록이 저장 위치 j = 12에 놓여 져야 한다면, 접근 경로 상에 있는 셀들(k = 13, 14, 그리고 15)이 점검되어야 한다.



Block moving directions			MP_{lj}
#	In	Out	
1	S → N	S ← N	{13, 14, 15}
2	S ↔ N	S ↔ N	{13, 14, 15}, {11}
4	S ↔ N W ↔ E	S ↔ N W ↔ E	{13, 14, 15}, {11}, {2, 7}, {17, 22}

Figure 2. A set of storage locations on the object moving path(j = 12)

본 논문은 각 단위 기간에 반출할 블록을 블록 적치장으로부터 먼저 출하 하고 반입할 블록을 블록 적치장에 저장하는 것으로 가정하기 때문에, 반출될 블록은 반입할 블록이 블록 적치장에 저장될 때 방해 블록이 될 가능성은 없다. 따라서 의사 결정 변수 X_{ijt}와 Y_{jt}는 <Figure 3>과 같은 구조를 갖는다. <Figure 3>은 시간 t = 1에 반입되어 시간 t = 5에 반출되는 블록에 대해 설명한다. 반입 블록을 블록 적치장에 저장하는 것과 관련하여, 시간 t = 1에서 의사 결정 변수 X_{ij1}는 식 (2)에 의해 제약 받는다. 반면에 반출 블록을 블록 적치장으로부터 출하하는 것과 관련하여 시간 t = 5에서 의사 결정 변수 X_{ij5}는 식 (6)에 의해 제약 받는다. 어떤 블록이 다른 반입 및 반출될 블록에 대해 방해 블록이 될 가능성이 있는 블록 적치장에 머무르는 기간 동안에 의사 결정 변수 Y_{jt}는 식 (5)에 의해 제약 받는다.

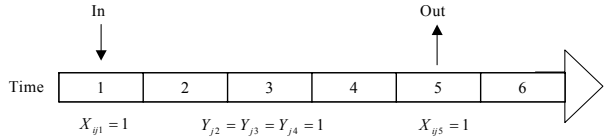


Figure 3. The structure of decision variables

본 논문에서 ABSLAP 문제에 대해 수립한 수리모형은 목적함수에 수학함수 min{ }와 곱하기 형태의 의사 결정 변수 X_{ijt}와 Y_{jt} 가 포함되어 있기 때문에 비선형 모형이 된다. 수학함수 min{ }을 선형모형으로 변환하기 위해 본 논문은 다음과 같은 식을 정의한다.

$$A_{lj} = \min_{l\in MD_j}\left\{\sum_{k\in MP_{lj}} X_{ijt}Y_{kt}\right\}$$

정의된 식을 이용하면 ABSLAP의 목적함수를 다음과 같이 선형모형으로 변환시킬 수 있다.

$$Minimize \sum_{t=1}^{T}\sum_{i\in OB_t}\sum_{j=1}^{J}\sum_{l\in MD_j} A_{lj} + \sum_{t=1}^{T}\sum_{i\in IB_t}\sum_{j=1}^{J}\sum_{l\in MD_j} A_{lj}$$

제약조건

$$\sum_{k\in MP_{ij}} X_{ijt}Y_{kt} \le A_{lj} \qquad \text{for } l\in MD_j,\ \forall j$$

제약조건을 추가로 더함으로써 새롭게 정의된 선형 목적함수는 본래의 비선형 목적함수 식 (1)과 동일하게 된다.

한편 곱하기 형태의 비선형은 Watters (1967)가 제안한 선형화 변환 기법을 이용하여 선형 모형으로 변환시킬 수 있다. 예를 들어, 변수가 0과 1의 값만 가지는 이차 방정식 $f(x, y)$을 고려해 보자. 이때 곱하기 형태의 xy를 새로운 변수 z로 대치하면 x와 y 값에 대응하는 z는 아래와 같은 값을 갖는다.

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

위와 같은 대응 관계는 아래의 식을 첨가함으로서 구해진다.

$$\begin{aligned} x+y-\ z &\le 1\\ -x-y+2z &\le 0\\ z &= 0 \text{ or } 1 \end{aligned}$$

(2) 제약식

(a) 단일 저장 위치 할당 제약식

$$\sum_{j=1}^{J} X_{ijt} = 1 \quad \text{for } i\in IB_t,\ \forall t \tag{2}$$

제약식 (2)는 반입되는 각 블록이 정확히 하나의 저장 위치에만 할당되도록 한다.

(b) 서로 다른 저장 위치 할당 제약식

$$X_{ijt}X_{i'jt} = 0 \quad \text{for } i < i',\ i \text{ and } i'\in IB_t,\ \forall j,\ \forall t \tag{3}$$

제약식 (3)은 반입되는 블록들이 서로 다른 저장 위치에 할당되도록 한다. 그리고 제약식 (3)에 포함된 곱하기 형태는 Watters(1967)가 제안한 선형화 변환 기법을 이용하여 선형 모형으로 변환될 수 있다.

(c) 빈 저장 위치 할당 제약식

$$X_{ijt}Y_{jt} = 0 \quad \text{for } i\in IB_t,\ \forall j,\ \forall t \tag{4}$$

제약식 (4)는 반입되는 각 블록이 빈 저장 위치에 할당되도록 한다. 여기서도 곱하기 형태는 Watters (1967)가 제안한 선형화 변환 기법을 이용하여 선형 모형으로 변환될 수 있다.

(d) 저장 위치 상태 제약식

$$\begin{aligned} Y_{jt'} &= \max\left\{Y_{jt'},\ X_{ijt}\right\}\\ &\text{for } t' = S_i+1,\cdots,F_i-1, i\in IB_t,\ \forall j,\ \forall t \end{aligned} \tag{5}$$

제약식 (5)는 반입된 블록 i가 블록 적치장에 머무르는 동안 저장 위치 j에 반입된 블록이 할당되어 있는 것을 나타낸다. 이 제약식은 max{ } 함수를 포함하고 있다. 그러나 의사 결정 변수가 모두 이진 변수이기 때문에 max{ } 함수는 다음과 같이 선형 모형으로 변환이 가능하다. 예를 들어, 곱하기 형태에 대한 선형화 변환 기법에서와 같이 max{x, y} 함수를 새로운 변수 z로 대치하면 x와 y 값에 대응하는 z는 아래와 같은 값을 갖는다.

x	y	z
0	0	0
0	1	1
1	0	1
1	1	1

위와 같은 대응 관계는 아래의 식을 첨가함으로서 구해진다.

$$\begin{aligned} x+y-2z &\le 0\\ -x-y+\ z &\le 0\\ z &= 0 \text{ or } 1 \end{aligned}$$

(e) 반출 블록 제약식

$$X_{ijF_i} = X_{ijt} \quad \text{for } i\in IB_t,\ \forall j,\ \forall t \tag{6}$$

제약식 (6)은 반출할 블록을 블록 적치장으로부터 출하할 때, 목적함수 식 (1)이 방해 블록의 수를 계산할 수 있도록 한다.

(f) 이진 정수 제약식

모든 의사 결정 변수는 0 또는 1의 값을 취한다.

3. 발견적 ABSLAP 알고리즘

블록 적치장을 운영할 때, 블록의 운반 작업이 필요한 경우는 3

가지가 있다: (1) 블록 적치장으로부터 반출할 블록을 출하 하는 경우, (2) 반출 블록의 이동 경로 상에 존재하는 방해 블록을 다른 빈 셀로 옮기는 경우, 그리고 (3) 블록 적치장에 반입 블록을 저장하는 경우. 이러한 세 가지 경우를 다루기 위해 발견적 ABSLAP 알고리즘은 두 개의 독립적인 하위-알고리즘으로 구성되어 있다: (1) 방해 블록 선정과 (2) 저장 위치 결정. 방해 블록 선정 하위-알고리즘은 반출할 블록을 출하하기 위해 블록 적치장에서 어떤 블록을 다른 빈 셀로 옮겨야 하는지를 결정한다. 그리고 저장 위치 결정 하위-알고리즘은 블록 적치장의 빈 셀을 반입 및 방해 블록에게 할당한다.

동적 상황을 다룰 수 있는 발견적 알고리즘의 장점 덕분으로 발견적 ABSLAP 알고리즘은 ABSLAP의 수리모형에서 세운 가정 사항들 가운데 하나를 제거한다. 즉, 수리모형은 방해 블록이 반입과 반출 작업이 완료된 후 원래의 위치로 되돌아가는 것으로 가정한다. 그러나 발견적 ABSLAP 알고리즘은 저장 위치 결정 하위-알고리즘을 이용하여 방해 블록을 새로운 빈 셀에 재 할당한다.

하위-알고리즘들을 설명하기 위해 본 논문은 블록 적치장으로부터 블록을 출하하는 예제를 이용한다. <Figure 4>는 하위-알고리즘의 모든 단계를 보여 준다. 우선 본 논문은 블록 이동 방향에 약간의 제한을 가정하고, <Figure 4(a)>는 이러한 가정 하에 블록 이동이 가능한 방향을 보여 주고 있다. 만약 어떤 방향으로의 블록 이동이 허용되지 않다면 하위-알고리즘은 제한된 방향과 관련된 단계를 생략한다.

3.1 방해 블록 선정 하위-알고리즘

블록 적치장으로부터 만기된 블록을 반출할 때 방해 블록이 발생할 수 있다. 방해 블록 선정 하위-알고리즘은 만기된 블록의 이동 거리를 최소화하고 방해 블록의 수를 최소로 갖는 반출 경로를 선택함으로써 방해 블록을 결정한다.

<Figure 4(b)>는 만기된 블록을 반출하기 위한 방향과 다른 빈 셀로 이동해야 할 방해 블록을 결정하는 방법을 보여준다. 셀 안에 있는 숫자는 블록이 반출되어야 하는 시점을 나타낸다. 현재 단위 시각이 5라고 가정하면, 음영 처리된 셀에 있는 두 개의 블록이 반출되어야 한다. 세 번째 열에 있는 만기된 블록은 반출하는데 전혀 방해물이 없다. 화살표는 블록의 이동 방향을 나타낸다. 하지만 네 번째 열에 있는 만기된 블록은 반출하는데 방해 블록이(즉, 점선으로 된 원으로 표시된 블록들) 존재한다. 화살표로 표시된 방향이 만기된 블록의 이동 거리를 최소화하고 방해 블록의 수를 최소로 갖는 경로 이다. 따라서 화살표로 표시된 방향에 있는 블록이 다른 빈 셀로 이동해

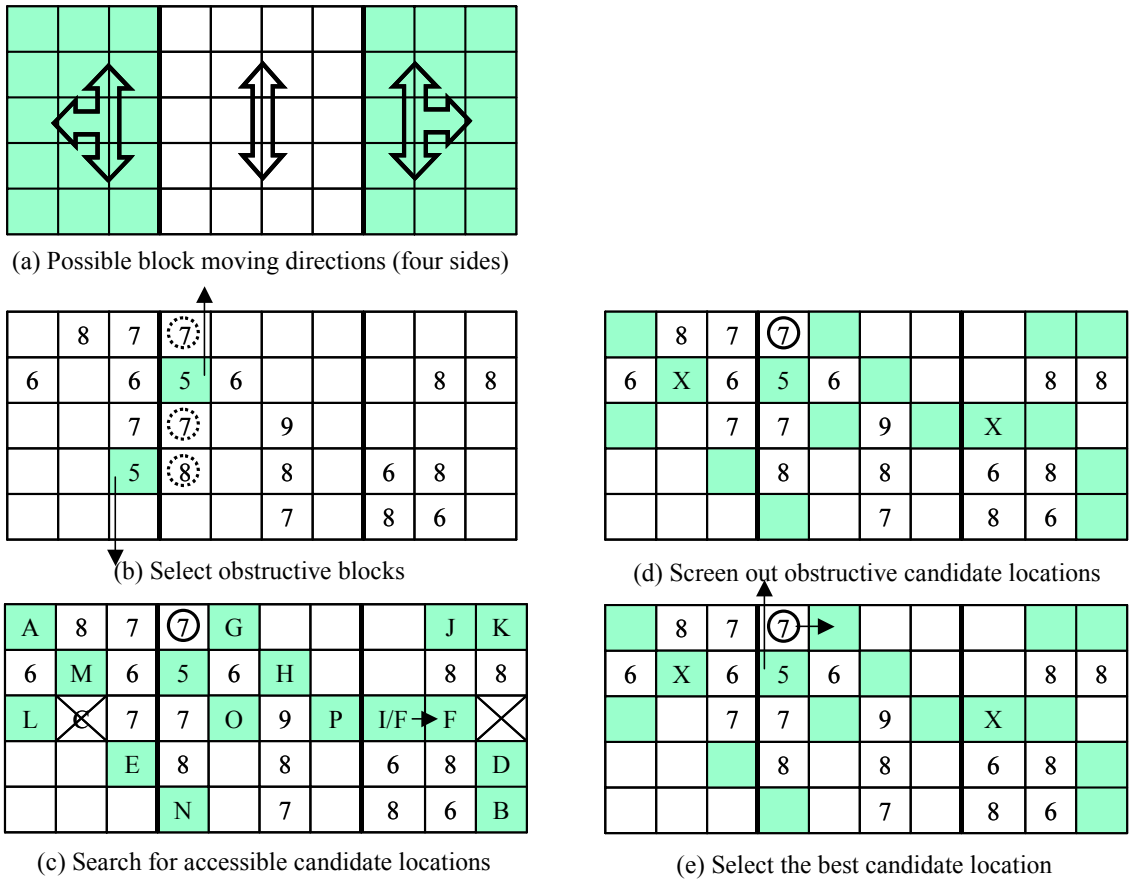


Figure 4. A summary of the heuristic ABSLAP algorithm

야 할 방해 블록으로 선정된다.

3.2 저장 위치 결정 하위-알고리즘

저장 위치 결정 하위-알고리즘은 두 가지 경우에 사용된다. 하나는 블록 적치장에 새로운 블록이 반입되는 경우이고, 다른 하나는 블록 적치장으로부터 만기된 블록을 반출시키는 경로에 방해 블록이 존재하는 경우이다. 새로이 반입되는 블록이나 방해 블록에게 빈 셀을 할당하기 위해 저장 위치 결정 하위-알고리즘은 다음과 같은 단계를 따른다.

단계 1 : 접근 가능한 후보 위치 탐색

후보 위치를 찾기 위해 이 단계는 모든 가능한 방향에서 모든 접근 가능한 후보 위치를 탐색한다.

단계 1.1 : W → E (E → W) 방향 검색

For the first row (i.e., depth = 1)
  If (North exit is open)
  {
    If (cell (1, 2) (cell (1, n-1)) is not empty and cell (1, 1)
      (cell (1, n)) is empty)
      cell (1, 1) (cell (1, n)) → accessible candidate location (e.g., A)
  }
  Else go to Otherwise

For the last row (i.e., depth = m)
  If (South exit is open)
  {
    If (cell (m, 2) (cell (m, n-1) is not empty and cell (m, 1)
      (cell (m, n)) is empty)
      cell (m, 1) (cell (m, n)) → accessible candidate location (e.g., B)
  }
  Else go to Otherwise

Otherwise, for each row
  The first empty cell located in front of the first occupied cell at the search direction → accessible candidate location (e.g., C & D)

  If (all cells are empty)
  {
    If (East (West) directional exit is possible)
      the middle cell → accessible candidate location
    Else the last cell → accessible candidate location (e.g., E & F)
  }

  If (accessible candidate location is open to the North or South exit)
    If (any cell in front of accessible candidate location is not open to both North and South exits)
    {
      Accessible candidate location is replaced by the first cell (e.g., F)
      Any cell in front of accessible candidate location cannot be candidate any more
    }

Step 1.2 : N → S (S → N) 방향 검색

For each column(width = 1 to n)
  The first empty cell located in front of the first occupied cell at the search direction → accessible candidate location (e.g., A, E & G - O)

  If (accessible candidate location crosses over any accessible candidate location)
    the accessible candidate location is cancelled (e.g., C)

  If (all cells are empty)
  {
    If (South (North) directional exit is possible)
    the middle cell → accessible candidate location (e.g., P)
    Else the last cell → accessible candidate location
  }

<Figure 4(c)>는 원으로 표시된 방해 블록을 다른 셀로 이동시키기 위한 본 단계를 보여준다. 음영 처리된 빈 셀들이 접근 가능한 후보 위치를 나타낸다.

단계 2 : 방해 후보 위치 제거

이 단계는 단계 1에서 탐색된 접근 가능한 후보 위치들 중에서 이동 대상인 블록이 위치하면 인접 셀에 있는 다른 블록이 이동 하는데 방해가 되는 후보 위치를 제거한다. <Figure 4(d)>는 방해 후보 위치를 X로 표시 하였다. 만약 원으로 표시된 방해 블록이 X로 표시된 셀로 이동한다면, 바로 오른쪽 셀과 아래 셀에 있는 블록은 이동 경로가 막혀 버린다. 즉, 이동 경로가 막힌 블록은 단위 시각 6에 블록 적치장으로부터 반출되어야 하는데 방해 블록은 단위 시각 7에 반출되는 것으로 되어 있다.

단계 3 : 최선의 후보 위치 선택

마지막 단계는 블록 이동 거리를 최소로 하는 최선의 후보 위치를 선정하는 것이다. <Figure 4(e)>는 원으로 표시된 방해 블록을 이동시킬 최선의 후보 위치가 바로 오른쪽 셀임을 보여준다. 화살표는 블록의 이동 방향을 나타낸다.

만일 블록 적치장에 여유 공간이 없음으로 인해 접근 가능한 후보 위치 탐색 단계에서 접근 가능한 후보 위치를 찾을 수 없을 경우, 이 블록 적치장에는 새로운 블록을 반입할 수 없다. 그러나 만기된 블록을 반출하기 위해 방해 블록들을 옮겨야 하는 경우는 방해 블록을 임시로 다른 장소로 옮겨 놓고, 만기된 블록을 반출한 후, 임시의 장소로 옮겨 놓은 방해 블록을 블

록 적치장에 재 반입시킨다.

만일 방해 후보 위치 제거 단계에서 모든 접근 가능한 후보 위치가 제거되면, 접근 가능한 후보 위치 탐색 단계에서 찾은 접근 가능한 후보 위치들 중에서 이동 거리를 최소로 하는 후보 위치를 최선의 후보 위치로 선정한다.

4. 컴퓨터 실험

4.1 수리모형과 발견적 ABSLAP 알고리즘의 비교

이 절에서는 시험 데이터를 이용하여 본 논문에서 수립한 수리모형과 교정한 발견적 ABSLAP 알고리즘의 성능을 검증해 본다. 시험 데이터는 현실 상황을 묘사하고 있으나, 수립한 수리모형에 기초한 최선의 해를 계산적으로 너무 무리한 부담을 갖지 않고 얻을 수 있도록 적절한 규모로 설계되었다. 시험 데이터에서 이용한 블록 적치장의 크기는 5×10이다. 전체 계획 기간은 100단위 기간으로 설정하였지만, 검증을 위한 시험 계획 기간은 블록 적치장이 안정적인 운영을 보인 50단위 기간 이후의 10단위 기간을 선택하였다. 각 단위 기간에 블록 적치장에 있는 블록의 평균수가 블록 적치장 최대 저장 용량의 70, 80, 90%가 되도록 반입할 블록을 무작위로 발생시켰다. 반입되는 모든 블록은 1에서 7단위 기간 사이의 임의 기간 동안 블록 적치장에 머물도록 무작위로 일정계획 되었다.

처음에 본 논문은 LINGO 소프트웨어가 제공하는 Branch-and-Bound 기법을 이용하여 시험 데이터에 대한 최적 해를 구하고자 하였다. 그러나 비록 블록 적치장의 크기를 비교적 적당하게 설정 하였지만 수립한 수리모형의 계산상 복잡성은 계획 기간에 따라 기하급수적으로 증가하였다. 시험 계획 기간을 한 번에 모두 고려하면서 시험 데이터에 대한 수리모형의 최적 해를 구할 수 없었다. 따라서 본 논문은 수립한 수리모형의 계산상 복잡성을 줄이기 위해 생산계획 분야에서 유용하게 이용되는 Rolling Through Time(RTT) 기법(Vollmann *et al*., 1988)의 사용을 고려하였다. RTT 기법을 이용하면 시험 데이터에 대한 수리모형의 최적 해를 얻을 수 있다는 보장은 할 수는 없지만, 최소한 최선의 해는 구할 수 있다.

Zhang *et al*.(2003)은 RTT 기법을 Rolling Horizon 접근 방법이라 불렀다. RTT 기법은 각 단위 기간에서 가까운 미래의 일정 기간에 대한 계획을 수립하고, 그 계획을 다음 단위 기간까지 실행한다. 그리고 RTT 기법은 최근의 정보를 받아들여 새로운 계획을 수립한다. 이러한 방식의 계획수립이 <Figure 5>에서 보여주는 바와 같이 계속적으로 진행된다. 이 기법은 선택된 계획 기간의 길이, 계산상의 부담 및 예측 능력 간에 절충관계를 가진다. 짧은 계획 기간을 사용하면 계산상의 부담은 줄어들지만, 미래에 대한 예측 능력은 떨어진다. 반면에 긴 계획 기간을 사용하면 계산상으로 실행 불가능할 수도 있으며, 너무 불확실한 정보를 포함할 수도 있다. 본 논문은 계획 기간을 5단위 기간으로 설정하고 수립한 수리모형을 시험 데이터에 대해 적용하였다.

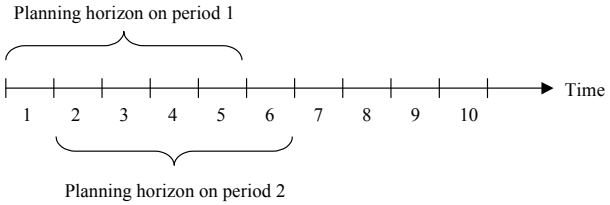


Figure 5. Rolling nature of planning horizon.

RTT 기법을 이용하여 시험 데이터에 대한 수리모형의 해를 구하는 과정에서 어떤 계획 기간에 대해서는 합리적인 CPU 시간 내에 최적 해를 구할 수 없었기 때문에 본 논문은 각 계획 기간에서 LINGO의 실행 시간을 2시간으로 제한하였다. 따라서 2시간 후 구해진 최선의 해를 다음 계획 기간에 사용 하였다. 또 다른 한편으로 본 논문은 교정한 발견적 ABSLAP 알고리즘을 이용하여 시험 데이터에 대한 해를 구하였다. 발견적 ABSLAP 알고리즘을 적용할 때, 블록 적치장에 블록을 반입하는 순서는 블록이 블록 적치장에 머무르는 기간의 내림차순을 따르게 했다. 즉, 블록 적치장에 오래 머무르는 블록을 우선적으로 블록 적치장에 반입하였다. <Table 1>은 목적함수 값 (OFV, objective function value) 관점에서 시험 데이터에 대한 수리모형에 기초한 최선의 해와 발견적 ABSLAP 알고리즘에 의한 결과를 비교한 내용을 보여준다.

Table 1. Comparison between the best solution and the heuristic algorithm

In-and-out directions	Stockyard workload(%)	Number of in and outbound blocks	OFV	
			Best	Heuristic
One side	70	167	0	0
		182	0	0
	80	198	9	8
		204	3	7
	90	227	42	45
		213	37	26
Two sides	70	167	0	0
		182	0	0
	80	198	0	0
		204	0	0
	90	227	12	4
		213	9	4
Four sides	70	167	0	0
		182	0	0
	80	198	0	0
		204	0	0
	90	227	8	2
		213	2	0

<Table 1>에서 수리모형에 기초한 최선의 해인 OFV와 비교해 볼 때, 발견적 ABSLAP 알고리즘은 아주 좋은 결과를 보여주고 있다. 블록의 반입 및 반출이 블록 적치장의 한쪽 면으로만 제한된 경우에서 발견적 ABSLAP 알고리즘은 수리모형에 기초한 최선의 해와 거의 비등한 성과를 보인다. 그러나 블록의 반입 및 반출이 블록 적치장의 2면 및 4면으로 확장되어 허용되는 경우에서는 발견적 ABSLAP 알고리즘이 수리모형에 기초한 최선의 해보다 훨씬 우수한 성과를 보여 주고 있다.

이렇게 발견적 ABSLAP 알고리즘이 수리모형에 기초한 최선의 해보다 더 좋은 성과를 보이는 이유를 밝혀내기 위해서 비교실험의 결과를 면밀히 분석해 본 결과, 본 논문은 그 이유 중의 하나로 발견적 ABSLAP 알고리즘의 동적 상황을 다룰 수 있는 능력에 주목한다. 다시 설명하면, 발견적 ABSLAP 알고리즘은 방해 블록을 능동적으로 다른 빈 셀로 재 위치시킴으로써 OFV를 감소시킬 수 있었다. 방해 블록을 능동적으로 다른 빈 셀로 재 위치시킬 수 있는 장점은 블록의 반입 및 반출이 블록 적치장의 2면 및 4면으로 확장되어 허용되는 경우에 더욱 확연해졌다.

위와 같은 발견적 ABSLAP 알고리즘의 성능에 대해 좀 더 분석하기 위해서 본 논문은 RTT 기법을 이용하여 시험 데이터에 대한 수리모형의 해를 구하는 과정에서 최적 해를 얻은 두 개의 계획기간을 선정하여 추가적인 비교실험을 실시하였다. 이러한 추가적인 비교 실험이 실시된 블록 적치장의 작업부하로 각 단위 기간에 블록 적치장에 있는 블록의 평균수가 블록 적치장 최대 저장 용량의 90%가 되는 경우를 선택하였다. <Table 2>는 OFV 관점에서 시험 데이터에 대한 수리모형의 최적 해와 발견적 ABSLAP 알고리즘에 의한 결과를 비교한 내용을 보여준다. 비록 추가적인 비교실험에서는 검증을 위한 시험 계획 기간이 5단위 기간으로 줄어들었지만, 수리모형의 최적 해와 발견적 ABSLAP 알고리즘에 의한 결과를 비교한 내용은 <Table 1>에 있는 수리모형에 기초한 최선의 해와 발견적 ABSLAP 알고리즘에 의한 결과를 비교한 내용과 유사한 결과를 보여주고 있다.

Table 2. Comparison between the optimal solution and the heuristic algorithm

In-and-out directions	Number of in and outbound blocks	OFV	
		Optimal	Heuristic
One side	109	28	11
	106	16	13
Twe sides	95	5	1
	106	9	2
Four sides	133	6	1
	125	3	1

4.2 블록 적치장 레이아웃의 영향

본 논문은 블록 적치장 레이아웃의 영향을 조사하기 위한 목적으로 다양한 크기의 블록 적치장을 대상으로 모의실험을 수행했다. 이러한 목적을 위해 고려한 블록 적치장의 크기는 5×5, 5×10, 5×15, 5×20, 5×25, 그리고 10×10이다. 이 모의실험에서 실험 계획 기간으로 전체 계획 기간 100 중에서 블록 적치장이 안정적인 운영을 보인 50단위 기간 이후의 50단위 기간을 선택하였다. 각 단위 기간에 블록 적치장에 있는 블록의 수가 평균적으로 블록 적치장 최대 저장 용량의 70, 80, 90%가 되도록 반입될 블록을 무작위로 발생시켰다. 반입되는 모든 블록은 1에서 7단위 기간 사이의 임의 기간 동안 블록 적치장에 머물도록 무작위로 일정계획 되었다. 그리고 54가지의 경우(레이아웃 6가지×부하 3가지×출입 허용면 3가지) 각각에 대해서 반입 및 반출되는 블록의 일정계획을 서로 다르게 만들어 발견적 ABSLAP 알고리즘을 5번씩 실행하였다.

<Figure 6>은 발견적 ABSLAP 알고리즘의 평균 OFV를 보여주고 있다. 블록 적치장의 부하 및 블록 출입 허용 면수 각각에 대해 발견적 ABSLAP 알고리즘의 평균 OFV는 블록 적치장의 폭이 증가함에 따라 감소하는 비슷한 추세를 보인다. <Figure 6>에서 보여 주는 결과로부터 블록을 반입 및 반출시킬 수 있는 경로의 수가 많으면 많을수록 방해 블록의 수가 감소한다는 것을 추측할 수 있다. <Table 3>은 이러한 추측을 확인시켜 준다. 블록 적치장 레이아웃 (5×20과 10×10) 이외의 모든 것이 동일한 조건 하에서 발견적 ABSLAP 알고리즘의 평균 OFV는 블록 적치장 레이아웃에 따라 아주 큰 차이를 보인다.

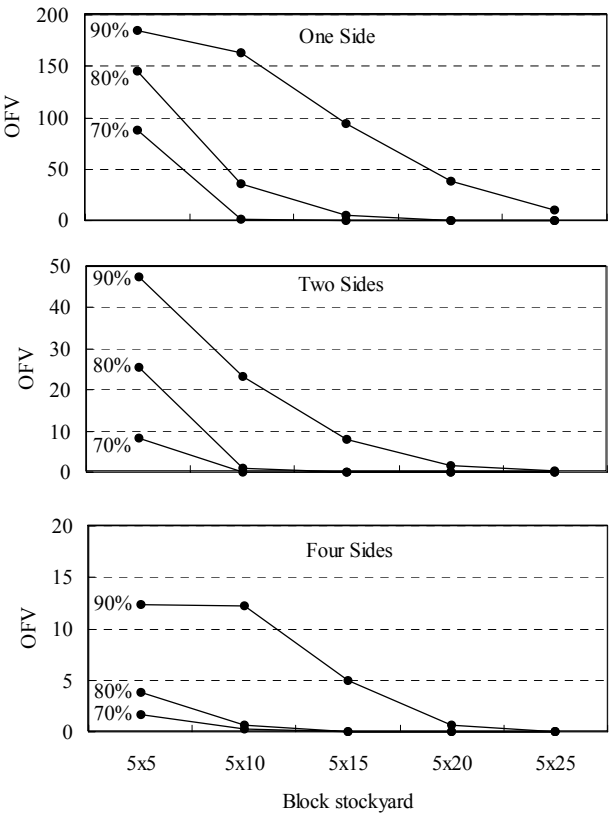


Figure 6. Effects of block stockyard layouts

Table 3. Influence of block stockyard layout

In-and-out directions	Block stockyard	OFV		
		70%	80%	90%
One side	5×20	0.0	0.4	38.4
	10×10	51.6	301.8	602.2
Two sides	5×20	0.0	0.0	1.6
	10×10	0.0	18.6	136.0
Four sides	5×20	0.0	0.0	0.6
	10×10	0.0	2.0	46.2

추가적으로 <Table 3>으로부터 블록 적치장에서 블록의 이동 거리가 증가함에 따라 방해 블록의 수가 증가한다는 것을 추론할 수 있다. 이러한 추론을 검증하기 위해 본 논문은 블록의 출입이 블록 적치장의 4면에서 허용되는 경우에 블록 이동 방향을 수정하여 발견적 ABSLAP 알고리즘을 수행하였다. 다시 설명하면, 본 논문에서 수행한 모의실험은 블록의 출입이 블록 적치장의 4면에서 허용되는 경우에 블록 이동은 <Figure 4(a)>에서 보여 주는 바와 같은 패턴으로 이루어졌다. 그러나 수정한 블록 이동 방향을 적용하는 경우에서는 <Figure 4(a)>에서 보여 주는 블록 이동 방향에 대한 모든 제약을 제거하였다. <Figure 7>은 블록의 출입이 블록 적치장의 4면에서 허용되고 블록 이동 방향에 대한 모든 제약을 제거한 경우, 블록 적치장 내에서 블록 적치장의 넓은 폭을 따라 이동하는 블록 이동 거리가 증가함에 따라 발견적 ABSLAP 알고리즘의 평균 OFV가 증가함을 보여준다.

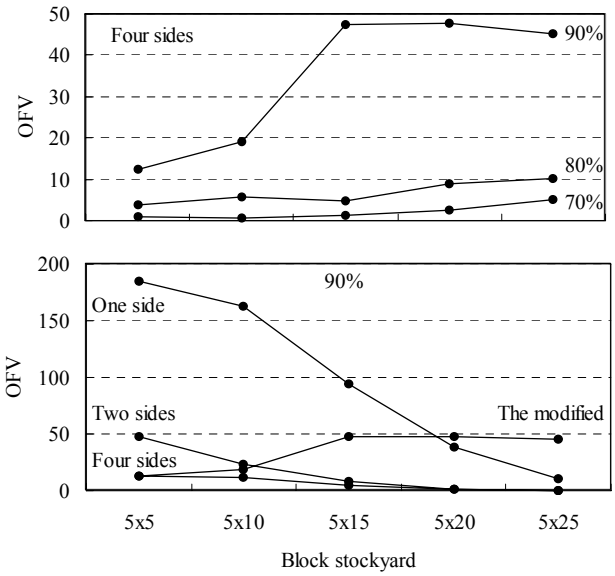


Figure 7. Influence of the block moving distance

4.3 생산 일정 계획 변동의 영향

안전 재고 수준이나 약속한 납기 준수율 등과 같이, 블록 적치장의 운영 성과는 불안정한 생산 일정 계획에 의해 크게 영향을 받을 수 있다. 발견적 ABSLAP 알고리즘을 사용할 경우, 블록 일정 계획의 변동에 의한 블록 적치장의 운영 성과에 대한 영향의 정도를 조사하기 위해 본 논문은 5×10 블록 적치장에 대해서 모의실험을 수행하였다. 실험조건은 블록 일정 계획이 변동 한다는 것을 제외하고는 제 4.2절에서의 조건과 동일하다.

본 논문에서는 블록의 계획된 반출 시기를 변동시킴으로써 블록 일정 계획이 불안정하게 되도록 만들었다. 즉, 각 단위 기간에 블록 적치장에 있는 블록의 10% (20%와 30%)를 무작위로 선택하여 반출 시기를 ±3 단위 기간 내에서 무작위로 변경시켰다. <Figure 8>은 블록 일정 계획 변동의 영향을 보여준다. 블록 적치장의 부하와 블록 출입 허용 면수 각각에 대해 발견적 ABSLAP 알고리즘의 평균 OFV는 블록 일정 계획 변동의 비율이 증가함에 따라 증가하는 비슷한 추세를 보인다. <Table 4>는 블록 일정 계획의 변동에 의해 야기된 실질적인 블록 적치장의 부하와 블록 적치장에 일시적으로 저장할 수 없는 초과된 블록의 수를 요약하여 보여준다.

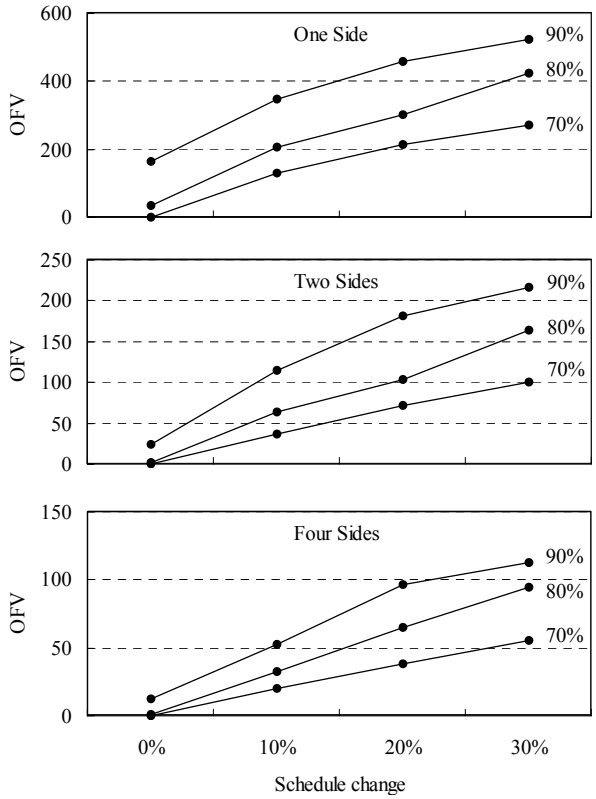


Figure 8. Impacts of production schedule instability

이 실험으로부터 생산 일정 계획의 안정성이 얼마나 중요한가를 쉽게 관찰할 수 있다. 90%의 블록 적치장 부하와 0%의 일정 계획 변동을 갖는 경우의 평균 OFV와 70 또는 80%의 블록 적치장 부하와 30%의 일정 계획 변동을 갖는 경우의 평균 OFV를 비교해 보자. 비록 블록 적치장의 부하가 더 높더라도 블록

Table 4. Changed block stockyard workload and block overflows

Block stockyard workload(%)	Block schedule change(%)	Chaned stockyard workload(%)	Block overflows	
			Max	Ave
70	10	70.43	0	0.00
	20	74.38	0	0.00
	30	76.03	5	0.02
80	10	80.86	0	0.00
	20	84.47	2	0.05
	30	88.15	7	0.16
90	10	90.54	5	0.15
	20	94.14	10	0.87
	30	96.35	12	1.67

일정 계획이 안정적인 경우가 불안정적인 경우보다 블록 적치장 운영에서 더 좋은 성과를 보여준다.

5. 결 론

본 논문은 Park and Seo(2006)가 처음으로 소개한 조선소에서의 조립블록 저장위치 할당문제(ABSLAP)에 대해 재조명해 보았다. 비록 Park and Seo(2006)가 다양한 상황에 대해 수행한 모의실험을 통하여 발견적 ABSLAP 알고리즘의 유용성을 입증하였지만, 그래도 발견적 ABSLAP 알고리즘을 수리모형 등과 같은 최적화 기법과 비교하여 성능을 검증해 보았으면 하는 아쉬움이 남아 있었다. 따라서 본 논문은 우선 ABSLAP에 대한 수리모형을 수립하였다.

그러나 ABSLAP에 대해 수립한 수리모형이 NP-hard 문제 범주에 속했다. 즉, 이 문제에 대한 계산적인 복잡성은 일정계획의 대상이 되는 블록의 수에 따라 기하급수적으로 증가한다. 따라서 이러한 문제에 대한 해를 지금까지 알려진 최적화 기법으로 합리적인 시간 내에 찾기란 어렵다. 이와 같은 상황은 큰 규모의 현실적인 문제를 풀기 위해 발견적 기법을 사용하는 것이 현명한 선택임을 의미한다.

본 논문은 기존의 발견적 ABSLAP 알고리즘을 보다 정교하게 교정하였고, 너무 무리한 계산적 부담 없이 수리모형으로도 풀 수 있는 적절한 크기의 문제에 대해서 교정한 발견적 ABSLAP 알고리즘의 성능을 검증하기 위해 ABSLAP의 수리모형을 비교 기준으로 활용하였다. 수리모형과 비교했을 때, 교정한 발견적 ABSLAP 알고리즘은 아주 좋은 결과를 보여 주었다. 동적 상황을 다룰 수 있는 발견적 알고리즘의 장점 덕분으로 교정한 발견적 ABSLAP 알고리즘은 수립한 ABSLAP의 수리모형보다 더 좋은(최소한 비등한) 성능을 보였다.

또한 본 논문은 교정한 발견적 ABSLAP 알고리즘을 사용할 경우, 블록 적치장의 운영 성과에 대한 블록 적치장 레이아웃의 영향과 블록 일정 계획의 변동에 의한 영향의 정도를 조사하기 위해 모의실험을 수행하였다. 모의실험 결과, 블록 적치장의 저장 용량이 동일할 경우 깊이가 낮고 폭이 넓은 블록 적치장 레이아웃이 블록 적치장 운영에 더 좋은 결과를 가져왔다. 그리고 비록 블록 적치장의 부하가 더 높더라도 블록 일정 계획이 안정적인 경우가 불안정적인 경우보다 블록 적치장 운영에서 더 좋은 성과를 보여 주었다.

참고문헌

Bazzazi, M., Safaei, N., and Javadian, N. (2008), A genetic algorithm to solve the storage space allocation problem in a container terminal, *Computers and Industrial Engineering*, doi : 10.1016/j.cie.2008.03.012.

Kozan, E. and Preston, P. (2006), Mathematical modeling of container transfers and storage locations at seaport terminals, *OR Spectrum*, **28**, 519-537.

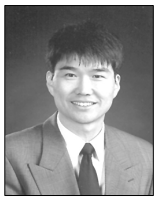
Park, C. and Seo, J. (2006), JIBUN (location) assignment algorithm for assembly blocks: a case of Hyundai Heavy Industries (in Korean), *IE Interfaces*, **19**, 160-167.

Preston, P. and Kozan, E. (2001), An approach to determine storage locations of containers at seaport terminals, *Computers and Operations Research*, **28**, 983-995.

Vollmann, T. E., Berry, W. L., and Whybark, D. C. (1988), *Manufacturing Planning and Control Systems*, Irwin: Illinois.

Watters, L. J. (1967), Reduction of integer polynomial programming problems to zero-one liner programming problems, *Operations Research*, **15**, 1171-1174.

Zhang, C., Liu, J., Wan, Y., Murty, K. G., and Linn, R. J. (2003), Storage space allocation in container terminals, *Transportation Research Part B*, **37**, 883-903.



박 창 규

고려대학교 산업공학과 학사
한국과학기술원 산업공학과 석사
University of Missouri-Columbia 산업공학과 박사
현재: 울산대학교 경영대학 경영학부 교수
관심분야: 공급사슬경영, 생산계획 및 통제 시스템

서 준 용

1994년 울산대학교 산업공학과 공학사
1996년 울산대학교 산업공학과 공학석사
2002년 울산대학교 산업공학과 공학박사
현재 : 울산대학교 외래강사
관심분야 : SCM, CIM, 데이터베이스 응용, e-Marketplace, 생산정보시스템