Journal of the Korean Institute of Industrial Engineers
Vol. 35, No. 2, pp. 129-140, June 2009.

평면적 저장 위치 할당 문제에 대한 유전자 알고리즘

박창규† · 서준용

울산대학교 경영대학

Genetic Algorithm of the Planar Storage Location Assignment Problem

Changkyu Park · Junyong Seo

College of Business Administration, University of Ulsan

This paper introduces the planar storage location assignment problem (PSLAP) that no research has attempted to mathematically solve. The PSLAP can be defined as the assignment of the inbound and outbound objects to the storage yard with aim of minimizing the number of obstructive object moves. The storage yard allows only planar moves of objects. The PSLAP usually occurs in the assembly block stockyard operations at a shipyard. This paper formulates the PSLAP using a mathematical programming model, but which belongs to the NP-hard problems category. Thus this paper utilizes an efficient genetic algorithm (GA) to solve the PSLAP for real-sized instances. The performance of the proposed mathematical programming model and developed GA is verified by a number of numerical experiments.

Keywords: Planar Storage Location Assignment Problem, Mathematical Programming Model, Genetic Algorithm, Assembly Block Stockyard

1. 서 론

본 논문은 평면적 저장 위치 할당 문제(PSLAP, planar storage location assignment problem)를 소개하고자 한다. 저장 공간 및 위치 할당 문제와 같은 유사한 형태가 항만 컨테이너 터미널 운영을 다루는 연구 분야에서 소개된 적이 있지만(Bazzazi *et al*., 2009; Kozan and Preston, 2006; Preston and Kozan, 2001; Zhang *et al*., 2003), 본 논문에서 소개하고자 하는 평면적 저장 위치 할당 문제를 수리적으로 접근하여 풀어보려는 시도를 한 연구는 아직 발표되어 있지 않다. PSLAP와 항만 컨테이너 터미널에서의 저장과 관련된 문제들은 기본적으로 비슷한 서비스, 즉 반입 및 반출되는 대상물(컨테이너와 조립블록)에게 일시적인 저장 장소를 제공하는 문제를 다룬다. 그러나 컨테이너는 항만 크레인에 의해 이동될 수 있기 때문에 운반 작업에서 PSLAP와 항만 컨테이너 터미널에서의 저장과 관련된 문제는 큰 차이를 보인다. 다시 말하면, 항만 컨테이너 터미널에서의 저장과 관련된 문제는 반입 및 반출되는 컨테이너를 3차원적으로(X축, Y축, Z축 방향) 이동시킬 수 있다. 그러나 PSLAP는 크레인 사용이 허용되지 않기 때문에 반입 및 반출되는 대상물을 2차원적으로만(X축 및 Y축 방향) 이동시킬 수 있다.

얼핏 보면, PSLAP는 단순히 대상물의 이동방향이 3차원에서 2차원으로 줄어든 항만 컨테이너 터미널에서의 저장과 관련된 문제의 특수한 문제로 오해될 수도 있다. 그러나 PSLAP와 항만 컨테이너 터미널에서의 저장과 관련된 문제는 운반 작업에서 큰 차이를 보이고, 각각 추구하는 목적함수가 다르기 때문에 항만 컨테이너 터미널에서의 저장과 관련된 문제(일반적으로 목적함수는 정박 중인 선박의 평균 체류기간 최소화)와는 다르게 PSLAP에 대한 문제를 명확히 정의하고 정형화하는 작업이 필요하다. 또한 PSLAP에 대한 해법도 새롭게 개발되어야 한다.

이 논문은 2008년 울산대학교의 연구비에 의해 연구되었음.
† 연락저자 : 박창규, 680-749 울산광역시 남구 대학로 102 울산대학교 경영대학, Tel : 052-259-2438, Fax : 052-247-7619,
E-mail : ckparkuou@ulsan.ac.kr
2008년 □월 □일 접수; 2008년 □월 □일 수정본 접수; 2008년 □월 □일 게재 확정.

PSLAP는 앞으로 설명하는 바와 같이 정형화할 수 있다. 먼저, <Figure 1>은 $m \times n$의 셀 (cell)로 구성된 저장소의 도식적인 모양을 보여 준다. 각 셀은 한 개의 대상물만을 저장할 수 있다. 즉, 한 셀에서 한 대상물 위에 다른 대상물을 쌓아 놓는 방식으로 여러 개를 동시에 저장하는 것은 허용되지 않는다. 따라서 저장소의 최대 저장 용량은 $m \times n$으로 계산된다. 저장소는 사각형의 모양을 갖는 서양 장기판(chessboard)과 흡사하다.

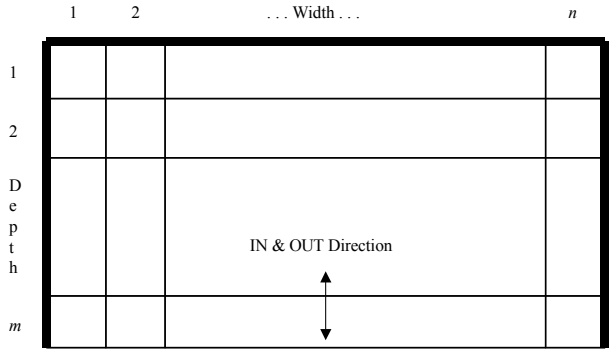


Figure 1. The schematic drawing of storage yard

이와 같은 저장소에서 대상물을 이동시키는 운반 작업을 효율적으로 수행하는 데 가장 큰 걸림돌은 대상물들을 단지 X축 및 Y축 방향으로만 이동시킬 수 있고 Z 방향으로는 이동시킬 수 없다는 제약이다. 따라서 저장소에 대상물을 반입 및 반출시킬 때, 어떤 대상물이 반입 및 반출되는 대상물의 이동 경로상에 존재한다면, 반입 및 반출되는 대상물에게 자유로운 이동 경로를 제공하기 위해 그 방해되는 대상물을 먼저 저장소로부터 빼내야 한다. 그리고 반입 및 반출 작업이 완료된 후, 저장소로부터 빼냈던 방해 대상물을 저장소의 원위치로 복원시킨다. 이러한 방해 대상물의 이동 작업은 비생산적인 활동으로 가급적 발생하지 않으면 좋을 것이다.

실제 상황에서 저장소는 시간에 구애받지 않고 연속적으로 운영된다. 그러나 본 논문은 연구를 위해 연속적인 시간을 이산적인 단위 기간(예, 하루)으로 나눈다. 각 단위 기간별로 저장소는 만기가 된 대상물을 반출시키고 반입되는 대상물을 저장한다. 그리고 반입되는 대상물은 저장소에 저장될 때 저장소로부터 반출될 시기가 미리 계획되어 진다. 이러한 상황에서 PSLAP는 방해 대상물의 이동횟수가 최소가 되도록 반입되는 대상물에게 저장 위치를 할당하는 것으로 정의될 수 있다.

저장소의 상황에 따라서 반입 및 반출되는 대상물은 저장소의 어떤 방향으로도 출입이 가능할 수 있을 것이다. 그러나 본 논문에서는 일반화 하는데 큰 손실이 없는 범위 내에서 문제의 복잡성을 가능한 한 단순화 시키기 위해 대상물이 저장소의 한쪽 방향으로만 출입이 이루어지는 경우를 다룬다. 반입 및 반출되는 대상물이 저장소의 임의의 어떤 방향으로도 출입이 가능한 경우는 향후 연구 과제로 남겨두고자 한다.

본 논문에서 소개하는 PSLAP는 조선 조립 블록을 취급하는 야외 적치장에서 발생된다. 선박 건조 과정에서 블록은 선박을 구성하는 기본 단위이다. 설계 단계에서 선박은 몇 개의 구역으로 나누어지고, 각 구역은 또 다시 블록으로 나누어진다. 내업 공장에서 철판 조각들을 용접하여 블록을 조립한 후 반출시키면, 반출된 블록들은 조선 현장에서 의장 및 도장 등과 같은 여러 공정들을 걸쳐 최종적으로 건조 도크에서 탑재되어 함께 용접된다.

조립 블록은 선박 건조 과정에서 재공품 재고에 해당되며 그 크기와 무게는 엄청나다. 조립 블록의 크기는 보통 가로 15미터, 세로 15미터, 그리고 높이 5미터 정도이며, 일부 조립 블록은 일반적인 크기보다 더 크다. 또한 조립 블록의 무게는 보통 100에서 300톤 사이이지만, 어떤 블록은 500톤을 넘는 것도 있다. 이처럼 조립 블록은 크고 무겁기 때문에 조립 블록 운영관리자는 운송과 저장에 특별한 주의를 기울인다. 조선소는 크고 무거운 조립 블록을 운반하기 위해 특별히 제작한 트랜스포터를 이용한다. 트랜스포터는 다방향 축의 바퀴를 65개나 달고 있는 운반 장비로, 거대하고 무거운 조립 블록을 운반하기 위해 조립 블록 아래로 들어가서 조립 블록을 들어 올리고 다른 장소로 이동할 수 있도록 유압 잭 리프트를 갖추고 있다.

특수 제작된 트랜스포터는 평평한 지지대로 블록을 들어 올려 한 번에 한 개의 블록을 운반한다. 블록 적치장에 조립 블록을 반입시킬 때, 빈 셀에는 트랜스포터가 도착하기 전에 2미터 높이의 지그 4개가 준비되어 있어야 한다. 그러면 조립 블록을 실은 트랜스포터는 지그들 사이로 들어가 조립 블록을 지그들 위에 얹어 놓고 자신만 빠져 나온다. 한편, 블록 적치장에서 조립 블록을 반출하려면, 먼저 트랜스포터가 해당 블록에 접근할 수 있는 경로를 확보해야 한다. 만약 어떤 블록이 접근 경로상에 놓여 있으면 트랜스포터는 반출할 블록에 접근하는데 방해가 되는 블록들을 다른 빈 공간으로 이동시켜야만 한다. 이러한 방해 블록들에 대한 운반 작업은 될 수 있으면 발생하지 않도록 해야 할 비생산적 활동이다. 블록 적치장을 운영하는 관리자 입장에서는 블록 위치를 신중히 결정하여 조립 블록 이동 횟수를 최소화하는 것이 대단히 중요하다.

PSLAP는 일반화된 할당 문제 형태로 모형화 할 수 있지만 이는 NP-hard 문제 범주에 속한다(Bazzazi *et al*., 2009; Preston and Kozan, 2001). 이 문제에 대한 계산적인 복잡성은 일정계획의 대상이 되는 대상물의 수에 따라 기하급수적으로 증가한다. 이러한 문제에 대한 해를 지금까지 알려진 최적해 기법(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 유전자 알고리즘이 아주 간단한 형태의 유전자 알고리즘 실험에서도 비교적 좋은 결과를 보인다고 보고되고 있기 때문에 효율적인 유전자 알고리즘을 활용하기로 한다.

본 논문의 앞으로의 구성은 다음과 같이 되어 있다. 먼저 제 2장은 PSLAP에 대해 수립한 수리모형에 관하여 설명한다. 그리고 제 3장은 본 논문에서 설계한 유전자 알고리즘을 자세히

설명한다. 다음으로 제 4장은 컴퓨터 실험 결과를 설명하고, 결론은 마지막 장에서 제시한다.

2. PSLAP의 수리모형

이 장에서는 다음의 가정 사항들을 기반으로 평면적 저장 위치 할당 문제(PSLAP)를 수리적으로 모형화 한다. PSLAP의 주 목표는 방해되는 대상물이 최소로 발생하도록 저장소에 대상물을 반입 및 반출시키는 것이다.

2.1 가정 사항

(1) 저장소에서 대상물은 일직선 방향으로만 움직일 수 있다. 즉, 대상물은 지그재그 형태로 움직일 수 없다.
(2) 각 단위 기간에서 반출할 대상물을 먼저 저장소로부터 출하시키고 난 후, 반입할 대상물을 저장소에 입하시킨다.
(3) 어떤 대상물을 반입 및 반출할 때 발생하는 방해 대상물은 우선 저장소로부터 빼낸다. 그리고 반입 및 반출 작업이 완료된 후, 저장소로부터 임시로 빼낸 방해 대상물을 원위치 시킨다.
(4) 저장소의 한 셀에는 한 대상물만 저장할 수 있다.

2.2 첨자 및 입력 모수

i : 대상물, $i = 1, 2, \cdots, I$
j : 저장 위치, $j = 1, 2, \cdots, J(J = m \times n)$
t : 시간(단위 기간), $t = 1, 2, \cdots, T$
IO_t : 시간 t에 반입될 대상물의 집합
OO_t : 시간 t에 반출될 대상물의 집합
S_i : 대상물 i의 반입 시간
F_i : 대상물 i의 반출 시간
m : 저장소의 깊이(행의 수)
n : 저장소의 폭(열의 수)

2.3 의사 결정 변수

X_{ijt} : 1, 반입될 대상물 i가 시간 t에 저장 위치 j에 할당될 경우 ; 0, 그 외의 경우
Y_{jt} : 1, 시간 t에 저장 위치 j에 대상물이 존재할 경우 ; 0, 그 외의 경우

2.4 수리모형

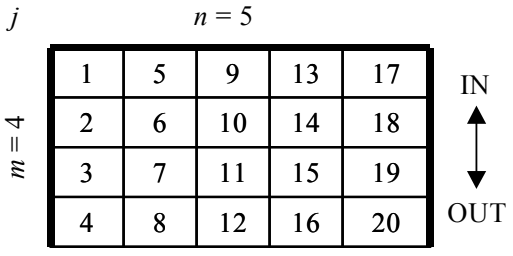
2.4.1 목적함수

PSLAP의 목적함수는 발생할 방해 대상물의 수를 최소화하는 것이다(수리모형이 좀 더 완벽해지려면 트랜스포터가 조립블록을 입출고시키기 위해서 블록 적치장을 움직인 거리를 고려해야 할 것이다. 하지만 실제 현장에서는 조립블록을 입출고시키기 위해서 적치장 내를 움직인 트랜스포터의 거리는 그렇게 심각하게 고려하지 않고 있다. 그 보다는 몇 개의 조립블록을 움직여야 하는가가 더욱 중요하다. 그 이유는 중후 장대한 조립블록을 이동시키기 위한 사전 준비 작업에 시간이 많이 소요되고, 또한 이동 작업에 안전상 많은 것이 요구되기 때문이다. 예를 들면, 적치장에 있는 어떤 블록을 바로 다음 셀로 옮기는 것도 비용이 많이 드는 작업이라고 볼 수 있다). 저장소 운영에서 방해 대상물이 발생할 수 있는 경우는 두 가지로 저장소로부터 반출할 대상물을 출하시킬 때와 저장소에 반입할 대상물을 입하시킬 때이다. 식 (1)은 각 경우에 발생할 방해 대상물의 수를 계산하는 방법을 보여준다.

$$Minimize \sum_{t=1}^{T}\sum_{i\in OO_t}\sum_{j=1}^{J} X_{ijt}\Big(\sum_{k=1}^{(m-j\%m)\%m} Y_{(j+k)t}\Big) + \sum_{t=1}^{T}\sum_{i\in IO_t}\sum_{j=1}^{J} X_{ijt}\Big(\sum_{k=1}^{(m-j\%m)\%m} Y_{(j+k)t}\Big) \quad (1)$$

식 (1)의 목적함수는 저장 위치 첨자 j가 저장소의 왼쪽 위 모서리에서 오른쪽 아래 모서리로 증가하는 순으로 번호가 부여되었다는 가정에 근거하고 있다. <Figure 2>는 크기가 4 × 5인 저장소와 방해 대상물의 수를 계산하기 위해 어떤 셀들을 점검해야 할 것인지를 설명하는 예제를 보여주고 있다. 예를 들어, 반출 또는 반입될 대상물이 저장 위치 $j = 6$에 놓여 져야 한다면 접근 경로 상에 있는 셀들이($j = 7$과 8) 점검되어야 한다. 여기서 연산자 %는 나머지 연산자로, 6%4 = 2를 의미한다.

본 논문은 각 단위 기간에 반출할 대상물을 저장소로부터 먼저 빼내고 반입할 대상물을 저장소에 저장하는 것으로 가정하기 때문에 반출될 대상물은 반입될 대상물이 저장소에 저장



j	$j\%m$	# of checking cells $(m-j\%m)\%m$	Checking cells
8	0	0	None
7	3	1	8
6	2	2	7, 8
5	1	3	6, 7, 8

Figure 2. The storage location numbering and checking

될 때 방해 대상물이 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 대상물에 대해 설명한다. 반입되는 대상물을 저장소에 저장하는 것과 관련하여 시간 t = 1에서 의사 결정 변수 X_{ijt}는 식 (2)에 의해 제약 받는다. 반면에 반출할 대상물을 저장소로부터 출하시키는 것과 관련하여 시간 t = 5에서 의사 결정 변수 X_{ij5}는 식 (6)에 의해 제약 받는다. 어떤 대상물이 다른 반입 및 반출될 대상물에 대해 방해 대상물이 될 수 있는 가능성이 있는 저장소에 머무르는 기간 동안에 의사 결정 변수 Y_{jt}는 식 (5)에 의해 제약 받는다.

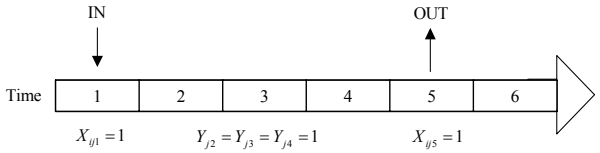


Figure 3. The structure of decision variables

본 논문에서 PSLAP 문제에 대해 수립한 수리모형은 목적함수에 의사 결정 변수 X_{ijt}와 Y_{jt}가 곱하기 형태로 포함되어 있기 때문에 비선형 모형이 된다. 그러나 이러한 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.

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

2.4.2 제약식

(a) 단일 저장 위치 할당 제약식

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

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

(b) 서로 다른 저장 위치 할당 제약식

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

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

(c) 빈 저장 위치 할당 제약식

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

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

(d) 저장 위치 상태 제약식

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

제약식 (5)는 반입된 대상물 i가 저장소에 머무르는 동안 저장 위치 j가 반입된 대상물에 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 대상물 제약식

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

제약식 (6)은 반출할 대상물을 저장소로부터 출하할 때 목적함수 식 (1)이 방해 대상물의 수를 계산할 수 있도록 한다.

(f) 이진 정수 제약식

모든 의사 결정 변수는 0 또는 1의 값을 취한다.

3. 유전자 알고리즘 적용

유전자 알고리즘은 많은 연구에 의해 그 효과성이 입증된 잘 알려진 메타-휴리스틱 기법이다(Bazzazi *et al*., 2009). 간단히 말해서, 유전자 알고리즘은 어려운 문제에 대해 최적의 해에 근접한 최선의 해를 구할 수 있는 발견적 기법이다. 이 기법은 진화와 유전의 원리에 기초하고 있다. 즉, 이 기법은 해의 집합인 모집단을 여러 세대에 걸쳐 진화시키면서 목적함수로 표현되는 적합도에 높게 반응하는 해를 찾는다. 본 논문에서 따른 유전자 알고리즘의 절차는 다음과 같다.

begin
  $t \leftarrow 1$;
  initialize Parents $P(t)$;
  evaluation $P(t)$;
  while (not termination condition) **do**
    recombine $P(t)$ to yield Offspring $C(t)$;
    evaluation $C(t)$;
    select $P(t+1)$ from $P(t)$ and $C(t)$;
    $t \leftarrow t+1$;
  end
end

본 논문은 문제를 풀기 위해 적용한 어떠한 유전자 알고리즘도 최소한 다음의 5가지 요소는 갖추고 있어야 한다고 한 Sarker and Newton(2002)의 주장을 따르고 있다.

(a) 해(또는 문제)의 표현
(b) 초기해 생성 방법
(c) 해의 적합도를 판단하는 평가함수
(d) 진화 과정에서 다음 세대를 생성하는 데 활용되는 유전 연산자
(e) 모수 값(모집단 크기, 유전 연산자에서 적용한 확률 등)

3.1 해의 표현

문제를 풀기 위해 유전자 알고리즘을 적용함에 있어, 첫 단계이면서 가장 중요한 단계는 해의 표현, 즉 염색체의 설계이다. 본 논문에서 다루는 PSLAP 문제를 풀기 위해서 고려해 볼 수 있는 염색체 설계 방법으로는 <Figure 4>에서 보여주는 바와 같이 4가지 정도가 있다. 염색체를 설계하는데 가장 간단한 방법은 <Figure 4>의 (a)에서 보여주는 바와 같이 목적함수의 의사 결정 변수를 이용하는 것이다. Bazzazi *et al*. (2009)이 염색체 설계에 이러한 접근 방법을 이용했다. 그러나 이 설계 방법을 PSLAP 문제에 적용하면 해의 표현에 컴퓨터 메모리를 너무 많이 낭비하는 결과를 초래한다.

염색체를 손쉽게 설계할 수 있는 두 번째 방법은 <Figure 4>의 (b)와 같이 저장소의 저장 위치를 이용하는 것이다. 이 설계 방법을 PSLAP 문제에 적용하면 염색체 크기가 계획 기간과 저장소 크기의 곱과 같기 때문에 해의 표현에 컴퓨터 메모리를 너무 많이 낭비하게 된다. 연구문헌에서 염색체 설계에 이

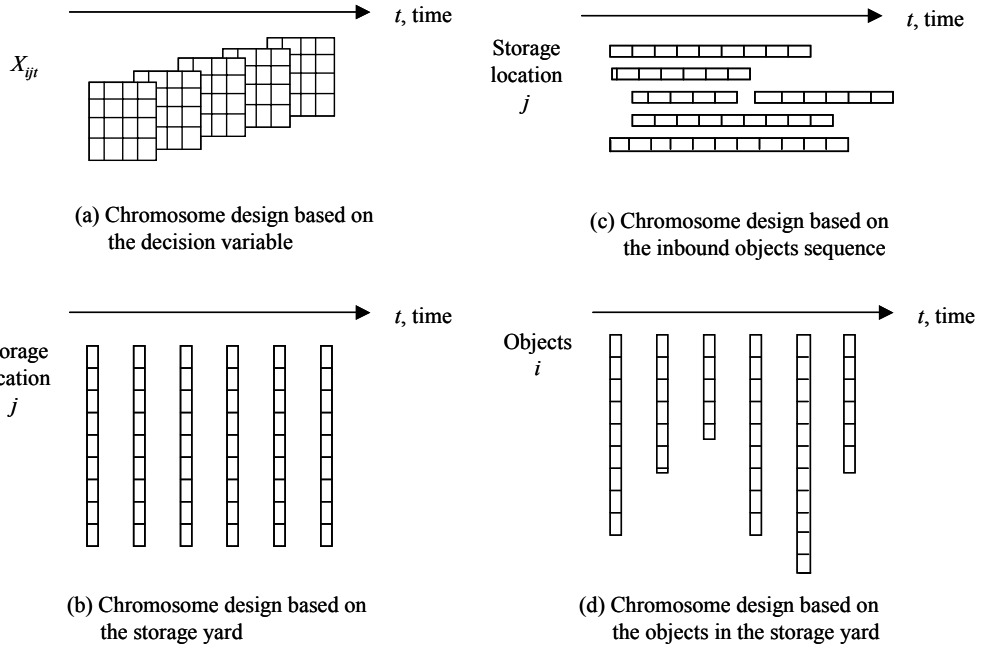


Figure 4. Chromosome designs

러한 접근 방법을 이용한 연구는 찾기가 어렵다.

염색체 설계의 세 번째 방법은 <Figure 4>의 (c)와 같이 반입되는 대상물의 순서를 이용하는 것이다. 염색체는 각 저장 위치에 반입되는 대상물의 순서를 나타낸다. 이러한 염색체 설계 방법은 정박 위치 할당 문제에 대한 연구 분야에 활용되어 졌다(Imai *et al*., 2003, 2006, 2007, 2008; Nishimura *et al*., 2001). PSLAP 문제를 푸는데 세 번째 염색체 설계 방법이 앞의 두 방법보다 훨씬 더 좋은 설계 방법이다. 그러나 이러한 염색체 설계 방법은 반입되는 대상물이 저장소의 최대 저장 용량을 초과하는 상황에 적합하다. 정박 위치 할당 문제에서는 일반적으로 선박이 정박 위치가 할당될 때까지 해상에서 대기하고 있다.

염색체 설계의 마지막 방법은 <Figure 4>의 (d)와 같이 저장소에 저장 중인 대상물을 이용하는 것이다. 이때, 염색체는 대상물이 저장된 저장 위치를 나타내는 정수형 문자열로 표현된다. 염색체의 길이는 전체 계획 기간 동안 각 단위 기간에 저장소에 저장 중인 대상물들을 모두 합한 수와 같다. 이 설계 방법은 컨테이너 저장 위치 할당 문제에 대한 연구에서 이용되어 졌다(Kozan and Preston, 1999, 2006; Preston and Kozan, 2001). 본 논문은 PSLAP 문제에 이 염색체 설계 방법이 가장 적합하다고 판단되어 이 방법을 사용한다.

3.2 초기해 생성

유전자 알고리즘에서 모집단의 크기를 결정하는 엄격한 규칙은 없다. 일반적으로 모집단의 크기가 크면 클수록 더 많은 컴퓨터 자원을 필요로 하지만 더 큰 다양성을 확보할 수 있다. 본 논문은 각 세대의 모집단 수를 30으로 설정한다. 초기 모집단은 염색체의 각 유전자에게 무작위로 정수(저장 위치)를 할당하여 생성한다. 그러나 초기해 생성 절차는 단위 기간별로 순차적으로 진행되어야 한다. 염색체는 전체 계획 기간 동안 저장소에 있는 대상물로 구성되어 있다. 반입된 대상물들은 임의의 기간 동안 저장소에 머물게 되고 이러한 대상물의 저장 위치가 변하면 안 된다. 따라서 초기해 생성 절차는 계획 기간 끝까지 순차적으로 각 단위 기간별로 반입될 대상물에게 빈 저장 위치를 무작위로 할당하게 된다.

3.3 적합도

PSLAP는 최소화 문제이다. 따라서 목적함수 값이 작으면 작을수록 적합도의 값은 높아진다. 본 논문은 목적함수의 역을 적합도로 정의한다.

3.4 유전 연산자

유전자 알고리즘의 3가지 기본적인 유전 연산자는 교차변이, 돌연변이 및 선택이다.

3.4.1 교차변이

교차변이는 주 유전 연산자이다. 교차변이는 한 번에 두 염색체의 유전자를 재결합하여 자식 염색체를 생성한다. 이러한 교차변이는 매력적인 해를 찾기 위해 다양한 방향으로 펼쳐나가는 검색을 가능하게 한다. 교차변이는 주로 one-cut-point나 two-cut-point 방법을 통해 수행된다. 본 논문에서는 two-cut-point 방법을 적용한다.

일반적으로 교차변이는 두 염색체 상에서 cut-point 방법을 적용하여 수행된다. 그러나 이러한 일반적인 교차변이를 PSLAP 문제에 그대로 적용할 수는 없고, 수정 작업이 필요하다. 즉, PSLAP 문제에 적용되는 염색체는 대상물과 시간으로 구성된 가상적인 이차원 구조를 갖게 되고 교차변이는 같은 단위 기간에 반입되는 대상물에만 적용할 수 있기 때문에 본 논문은 교차변이를 수행하기 위해 먼저 단위 기간을 무작위로 선택한다.

다음으로 선택된 단위 기간에서의 저장소 상황을 나타내는 임시 염색체를 만든다. 임시 염색체는 선택된 단위 기간 이전에서부터 존재했던 대상물의 저장 위치는 제외시키고 선택된 단위 기간에 반입되는 대상물의 저장 위치와 빈 저장 위치만 포함하도록 구성된다. 본 논문은 임시 염색체에 two-cut point 교차변이를 수행한다. 여기서 임시 염색체를 이용하지 않으면 교차변이는 후속되는 계획 기간에서 저장소에 저장되어 있는 대상물의 저장 위치를 뒤섞어 혼란시키는 심각한 문제를 야기한다. 컨테이너 저장 위치 할당 문제에 대한 연구는(Kozan and Preston, 1999, 2006; Preston and Kozan, 2001) 단순히 단일 기간만을 다루었기 때문에 임시 염색체에 대한 고려를 하지 않았다.

교차변이는 제약식 (2)를 만족시키지 못하는 실행 불가능한 자식 염색체를 만들 수도 있다. 다시 말해서, 자식 염색체가 반입되어야 하는 대상물을 포함하지 않거나, 또는 반입되는 대상물을 중복으로 포함할 수 있다. 이러한 실행 불가능한 자식 염색체를 실행 가능한 염색체로 전환하여 주기 위해 본 논문에서는 다음과 같은 교차변이 교정절차를 밟고 있다. <Figure 5>는 교차변이에 의해 만들어진 새로운 자식 염색체의 예제를 보여주고 있다. 우선, 상호 교환될 하부 문자열(진하게 칠해진 하부 문자열)이 two-cut-point 방법에 의해 무작위로 결정된다. 염색체 A와 B 사이에 선택된 하부 문자열이 교환된 후 염색체 C'과 D'가 임시의 자식 염색체로 만들어 진다. 이들 염색체는 실행 불가능하다. 예를 들어, 염색체 C'은 반입될 대상물 3, 5, 6, 8, 12를 두 번 포함하고 있고, 반입될 대상물 7, 10, 11은 포함하고 있지 않다. 다음으로 자식 염색체를 실행 가능하게 만들기 위해 다음에 설명하는 바와 같이 추가적인 상호 교환 작업을 수행한다. 본 논문은 <Figure 5>에서 보여주는 괄호로 쌓인 번호를 따라가며 교차변이 교정절차를 설명한다.

(1) 염색체 C'에서 처음으로 두 번 나타나는 반입될 대상물은 3이다. 이 유전자에 대응하는 위치에 있는 염색체 D'의 반입될 대상물은 7이다. 이제 염색체 C'에 반입될 대상물 7이 존재하거나, 또는 존재하지 않는 두 가지 가능성이 발생한다. 이 예제에서는 염색체 C'에 반입될 대상물 7이 존재하지 않기 때

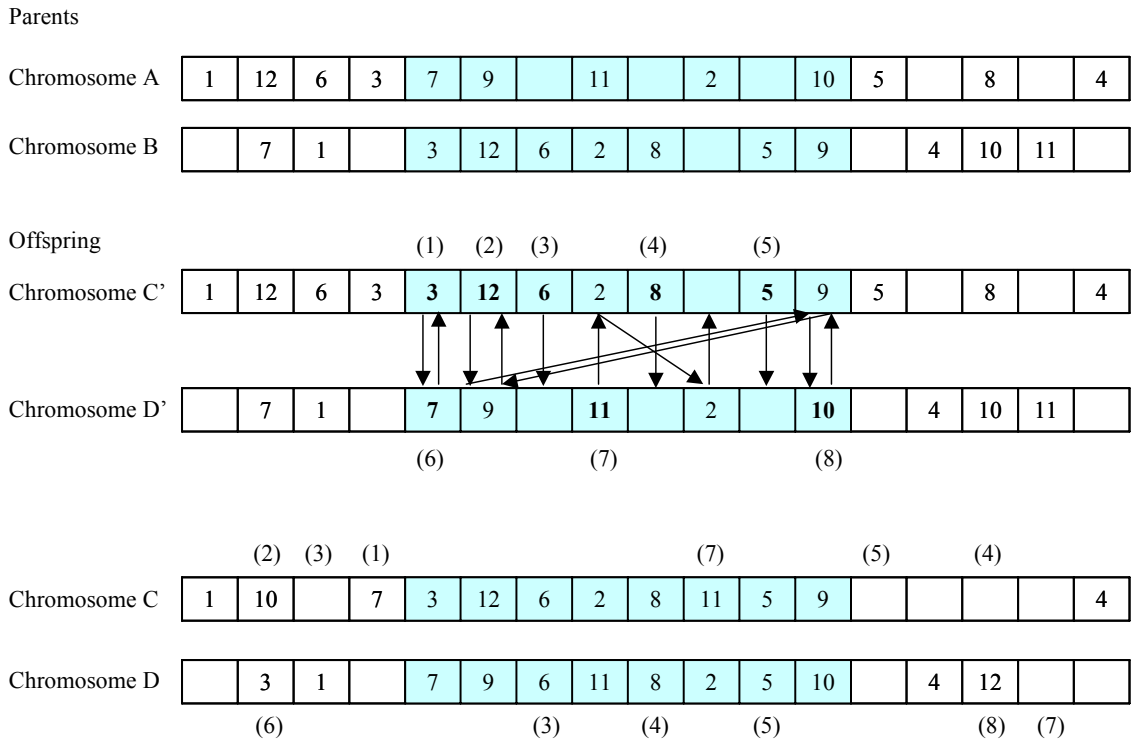


Figure 5. Example of crossover processing

문에 상호 교환된 하부 문자열의 바깥 부분에 있는 반입될 대상물 3을 7로 대치한다.

(2) 염색체 C'에 두 번 포함된 반입될 대상물 12와 대응되는 염색체 D'의 유전자 위치에 있는 반입될 대상물은 9이다. 염색체 C'에 반입될 대상물 9가 존재하므로 염색체 C'에서 반입될 대상물 9가 있는 유전자 위치를 찾는다. 염색체 C'에 있는 반입될 대상물 9와 대응되는 염색체 D'의 유전자 위치에는 반입될 대상물 10이 있다. 염색체 C'에 반입될 대상물 10이 존재하지 않으므로 상호 교환된 하부 문자열의 바깥 부분에 있는 반입될 대상물 12를 반입될 대상물 10으로 대치한다.

(3) 염색체 C'에 두 번 포함된 반입될 대상물 6과 대응되는 염색체 D'의 유전자 위치에는 반입될 대상물이 존재하지 않는다. 이러한 경우, 상호 교환된 하부 문자열의 바깥 부분에 있는 반입될 대상물 6과 염색체 D'에 있는 빈 저장 위치를 서로 교환한다.

(4)-(8) 나머지 교차변이 교정절차는 앞에서 설명한 절차의 반복이다.

마지막으로 선택된 단위 기간에 대해 교차변이 교정절차를 실시한 결과에 의해 영향을 받는 후속하는 기간들에 대해 약간의 교정 작업을 수행한다. 이 교정 작업에서는 후속하는 기간에 대해 변경된 반입될 대상물의 위치를 반영한다.

3.4.2 돌연변이

돌연변이는 사용자가 명시한 확률인 '돌연변이율'만큼 유전자의 값을 변경함으로써 염색체에게 임의의 변이를 안겨 준다. 교차변이에서와 같이, 본 논문은 선택된 단위 기간 이전부터 존재했던 대상물들의 저장 위치는 제외시킨 저장소의 상황을 나타내는 임시 염색체에 돌연변이를 수행한다. 돌연변이는 무작위로 선택한 두 저장 위치를 서로 바꾸고 돌연변이에 의해 영향을 받는 후속의 기간들에 대해 약간의 교정 작업을 수행한다.

3.4.3 선택

유전자 알고리즘의 기본 원리는 다윈의 자연 선택에 기초하고 있다. 선택은 유전자 알고리즘에게 강력한 추진력을 제공한다. PSLAP 문제에서 본 논문은 선택 단계와 관련된 두 가지 이슈, 즉 표본공간(sampling space)과 표본기법(sampling mechanism)을 고려한다.

선택 절차는 다음 세대를 위해 새로운 모집단을 생성한다. 이때 선택 절차는 부모 염색체와 자식 염색체 모두를 포함한 집합에 기초할 수도 있고, 또는 이들 일부분에 기초할 수도 있다. 따라서 이러한 과정은 표본공간에 대해서 고려하게 만든다. 본 논문은 정규 표본공간보다는 확장된 표본공간을 적용한다. 확장된 표본공간은 부모 염색체와 자식 염색체 모두를 포함하는 표본공간으로 확장된 표본공간의 크기는 모집단의 크기와 자식 집단의 크기를 합한 규모이다. 반면에 정규 표본공간은 모집단의 크기만큼의 규모이고, 자식 염색체 모두와

Table 1. Test sets for the performance comparison

	Test set			
	Test set 3×5		Test set 5×10	
	3 × 5_1	3×5_2	5×10_1	5×10_2
Test period	10 unit periods			
Storage yard size	3×5		5×10	
Storage yard workload	75% on average		80% on average	
Number of in and outbound objects	72	66	198	204
Stay period of inbound object	1～5unit periods		1～7unit periods	

부모 염색체의 일부분만을 포함하고 있다. 선택이 확장된 표본공간에서 수행될 때 생존을 위해 경쟁하는 부모 염색체와 자식 염색체의 생존 기회는 모두 동일하다. 확장된 표본공간의 명백한 장점은 다음과 같은 것이다. 즉, 유전자 알고리즘의 성능은 교차변이와 돌연변이의 율을 증가시킴으로서 개선시킬 수 있다. 그러나 높은 변이율은 너무 심한 무작위 변동을 초래할 수 있다. 만약 선택이 확장된 표본공간에서 수행된다면 이러한 걱정을 할 필요가 없어진다(Gen and Cheng, 1997).

표본기법은 표본공간으로부터 어떻게 염색체를 선택할 것인가 하는 문제와 관련이 있다. 본 논문은 확정적 표본(deterministic sampling)에 속하는 절단 선택(truncation selection) 기법을 활용한다. 절단 선택 기법은 모든 염색체들을 적합도에 따라 순위를 매기고 모집단 크기만큼만 상위에서 잘라서 선택하는 것이다. 이 과정에서 본 논문은 중복된 염색체가 모집단에 포함되지 못하도록 한다. 이러한 전략을 사용하는 이유는 다음과 같은 두 가지 때문이다 : (a) 우수한 염색체들이 너무 많이 복사되어 모집단을 지배하는 것을 막음으로서 너무 빨리 지엽적 최적해에 조기 수렴하는 것을 방지할 수 있다 ; (b) 모집단의 다양성을 유지하여 다음 세대를 생성할 때 유전적 검색에 보다 많은 정보를 제공할 수 있다(Gen and Cheng, 1997).

3.5 모수 값들의 요약

본 논문에서 수행한 모든 유전자 알고리즘 실험은 다음과 같은 특성을 갖는다.

- 교차변이율(= 자식 개체수 / 모집단 크기) : 0.3
- 돌연변이율(= 전체 유전자 수에 대한 비율) : 0.1
- Two-cut-point 교차변이와 유전자 교환 돌연변이
- 모집단 크기 : 30
- 각 실험에서의 세대 수 : 1000(일부는 2000)
- 독립적 실험횟수 : 5

4. 컴퓨터 실험

4.1 수리모형과 유전자 알고리즘의 성능 검증

이 절에서는 <Table 1>에서 보여 주는 2개의 시험 데이터를 이용하여 본 논문에서 제안한 수리모형과 개발한 유전자 알고리즘의 성능을 검증해 본다. 우선, 시험 데이터 3 × 5는 현실 상황을 묘사하고 있으나 제안한 수리모형이 계산적으로 너무 무리한 부담을 갖지 않고 풀 수 있도록 작은 규모로 설계되었다. 시험 데이터에서 저장소의 크기는 3 × 5이다. 시험 기간은 10 단위 기간이고, 각 단위 기간에 저장소에 있는 대상물의 수가 저장소 최대 저장 용량의 70%와 80% 사이에 있도록(평균적으로 75%) 반입할 대상물을 무작위로 발생시켰다. 반입되는 모든 대상물은 1에서 5 단위 기간 사이에 임의의 기간 동안 저장소에 머물도록 무작위로 일정계획 되었다.

시험 데이터 3 × 5에 대한 최적의 해는 LINGO 버전 10.0 소프트웨어가 제공하는 Branch-and-Bound 기법을 이용하여 풀었다. 한편 본 논문에서 제안한 유전자 알고리즘은 MS-Access 데이터베이스와 Visual Basic을 이용하여 구현하였다. 시험 데이터 3 × 5에 있는 각 시험 데이터에 대해서 5회씩 유전자 알고리즘의 컴퓨터 실험을 실시하였고, 각 실험마다 무작위로 생성한 초기 염색체로 구성된 서로 다른 모집단(모집단 크기 = 30)을 사용하였다. 각 실험에서 유전자 알고리즘의 컴퓨터 진행을 1000세대까지 실시하였다. <Table 2>는 목적함수 값(OFV,

Table 2. Comparison between the optimal solution and GA for the test set 3 × 5

Test set	Inbound and outbound	OFV						
		Optimal	Gentic algorithm					
			1	2	3	4	5	Average
3×5_1	72	0	2	1	1	2	1	1.40
3×5_2	66	0	1	2	2	1	1	1.40

objective function value) 관점에서 시험 데이터 3 × 5에 대한 수리모형에 의한 최적해와 유전자 알고리즘의 결과를 비교한 내용을 보여주고 있다.

수리모형에 의한 최적해 OFV와 비교해 볼 때, 본 논문에서 개발한 유전자 알고리즘은 아주 좋은 결과를 보여주고 있다. OFV에 대한 비교 결과, 유전자 알고리즘이 PSLAP 문제를 해결하기 위한 좋은 대안이 될 수 있음을 알 수 있다. <Figure 6>은 시험 데이터 3 × 5에 대해 500세대 동안 유전자 알고리즘의 평균 OFV가 어떻게 수렴하는지를 보여주고 있다. 본 논문은 1000세대까지 유전자 알고리즘을 수행했지만, 유전자 알고리즘의 OFV가 300세대 정도에서 이미 충분히 수렴되었기 때문에 <Figure 6>은 오직 500세대까지 만의 결과를 보여준다.

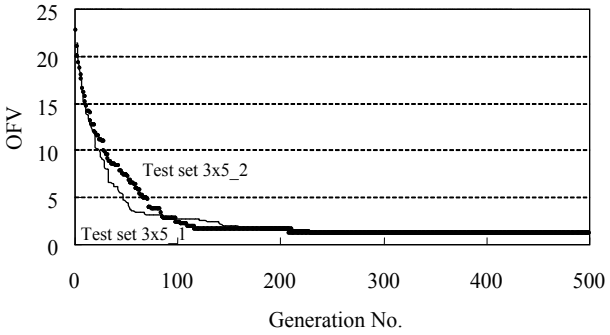


Figure 6. The OFV convergence of GA for the test set 3 × 5

다음으로 개발한 유전자 알고리즘이 계산 시간 관점에서 얼마나 효율적으로 시험 데이터를 푸는지 검정하기 위해 시험 데이터 5 × 10은 실제 저장소의 크기 상에서 현실 상황을 묘사하도록 설계되었다. 시험 데이터에서 저장소의 크기는 5 × 10이다. 시험 기간은 10단위 기간이고, 각 단위 기간에 저장소에 있는 평균 대상물의 수가 저장소 최대 저장 용량의 80%가 되도록 반입할 대상물을 무작위로 발생시켰다. 반입되는 모든 대상물은 1에서 7 단위 기간 사이에 임의의 기간 동안 저장소에 머물도록 무작위로 일정계획 되었다.

처음에 본 논문은 LINGO 버전 10.0 소프트웨어가 제공하는 Branch-and-Bound 기법을 이용하여 시험 데이터 5 × 10에 대한 최적 해를 구하고자 하였다. 그러나 저장소의 크기가 증가함에 따라, 그리고 시험 기간에 따라 제안한 수리모형의 계산상 복잡성은 기하급수적으로 증가하였다. 전체 시험 기간을 한 번에 모두 고려하면서 시험 데이터 5 × 10에 대한 수리모형의 최적 해를 구할 수 없었다. 따라서 본 논문은 제안한 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 데이터 5 × 10에 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 7>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 제안한 수리모형을 시험 데이터 5 × 10에 대해 적용하였다.

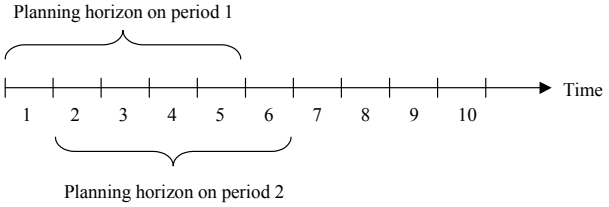


Figure 7. Rolling through time

RTT 기법을 이용하여 시험 데이터 5 × 10에 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 사용하였다. 또 다른 한편, Intel(R) Pentium(R) 4 CPU 3.06GHz and 992MB RAM의 용량을 갖춘 퍼스널 컴퓨터를 이용, 시험 데이터 3 × 5에 적용한 것과 동일한 실험 조건 하에서 개발한 유전자 알고리즘으로 시험 데이터 5 × 10을 풀었다. <Table 3>은 OFV와 CPU 시간 관점에서 시험 데이터 5 × 10에 대한 수리모형에 기초한 최선의 해와 유전자 알고리즘의 결과를 비교한 내용을 보여주고 있다.

시험 데이터 3 × 5에 대한 수리모형에 의한 최적해와 유전자 알고리즘의 비교 결과처럼, 시험 데이터 5 × 10에 대한 수리모형에 기초한 최선의 해와 유전자 알고리즘의 OFV를 비교해 볼 때, 유전자 알고리즘이 PSLAP 문제를 해결하기 위한 좋은 대안이 될 수 있음을 알 수 있다. 그리고 시험 데이터 5 × 10을 풀기 위해 걸린 유전자 알고리즘의 CPU 시간도 적절함을 볼 수 있다. <Figure 8>은 시험 데이터 5 × 10에 대해 1000세대 동안 유전자 알고리즘의 평균 OFV가 어떻게 수렴하는지를 보여주고 있다.

4.2 추가적인 GA 실험

다음으로 본 논문은 좀 더 크면서 현실적인 크기의 저장소에 대해 개발한 유전자 알고리즘을 적용해 보는 실험을 수행하였다. 이 실험을 위해 크기가 5×10과 5×15인 저장소가 고려

Table 3. Comparison between the best solution and GA for the test set 5 × 10

Test set	Inbound and outbound	Method		OFV	CPU time (second)
5×10_1	198	Math based model	PH = 5	9	N/A
		GA	1	5	78
			2	10	84
			3	6	84
			4	9	77
			5	11	78
			Average	8.20	80
5×10_1	204	Math based model	PH = 5	3	N/A
		GA	1	3	80
			2	5	84
			3	9	86
			4	7	89
			5	8	76
			Average	6.40	83

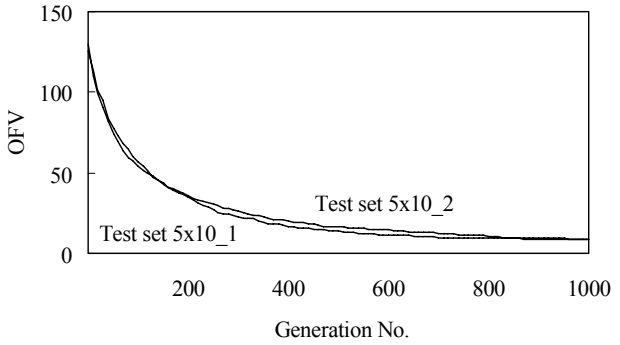


Figure 8. The OFV convergence of GA for the test set 5 × 10

되었고, 전체 계획 기간은 10단위 기간으로 설정되었다. 각 단위 기간에 저장소에 있는 대상물의 수는 평균적으로 저장소 최대 저장 용량의 70, 80, 90%가 되도록 반입될 대상물이 무작위로 생성되었다. 반입되는 모든 대상물은 1에서 5 단위 기간 사이에 임의의 기간 동안 저장소에 머물도록 무작위로 일정계획 되었다. 앞에서 설명한 시험 데이터에 대한 유전자 알고리즘 실험과 같이, 6가지 경우(2종류의 저장소 × 3종류의 저장소 부하)에 대해서 5회씩 유전자 알고리즘의 컴퓨터 실험을 실시하였고, 각 실험마다 무작위로 생성한 초기 염색체로 구성된 서로 다른 모집단(모집단 크기 = 30)을 사용하였다. 각 실험에서 유전자 알고리즘의 컴퓨터 진행을 1000세대까지 실시하였다.

<Figure 9>은 유전자 알고리즘의 결과인 평균 OFV의 수렴 정도를 1000세대까지 보여주고 있다. 각 저장소의 크기에서 유전자 알고리즘의 평균 OFV 수렴은 비슷한 결과를 보인다. <Figure 9>로부터 저장소에 많은 대상물이 존재하여 복잡하면 할수록 방해되는 대상물의 수가 증가한다는 것을 알 수 있다. 즉, 방해 대상물의 수와 저장소의 작업부하는 양의 상관관계를 보인다. 그리고 <Figure 9>는 정교한 알고리즘을 사용함으로써 비생산적인 방해 대상물의 이동횟수를 획기적으로 줄일

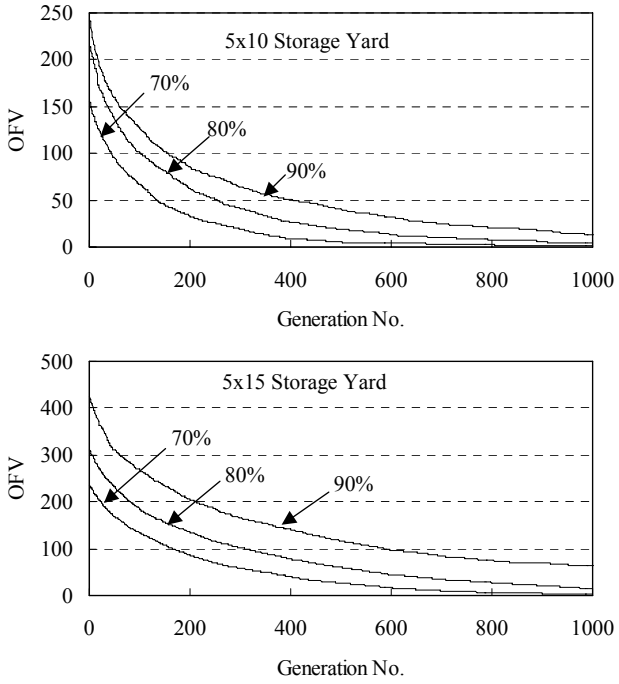


Figure 9. The OFV convergence of GA for the additional GA experiments

수 있음을 보여주고 있다. <Table 4>는 추가적인 GA 실험의 결과를 요약하여 보여주고 있다. 1000세대까지의 실행에서 충분히 수렴하지 않는 경우에 대해서는 유전자 알고리즘 실험을 2000세대까지 증가시켰다. 그 결과는 <Table 4>에서 괄호 안에 나타내었다.

4.3 교차변이에서 임시 염색체에 대한 비교

교차변이를 수행하기 위해 본 논문에서 개발한 유전자 알고

Table 4. Results of GA computations for the additional GA experiments

Storage yard	Workload (%)	Inbound and outbound	OFV					
			1	2	3	4	5	Average
5×10	70	98	1	1	1	1	3	1.4
	80	119	5	2	2	7	6	1.4
	90	151	15(7)	17(10)	15(8)	11(4)	9(4)	13.4(6.6)
5×15	70	147	3	2	7	5	2	3.8
	80	165	23(7)	10(3)	16(5)	13(4)	11(3)	14.6(4.4)
	90	196	60(15)	73(22)	55(15)	81(24)	47(15)	63.2(18.2)

리즘은 선택된 단위 기간의 저장소 상황을 나타내는 임시 염색체를 이용하였다. 임시 염색체를 구성하는 방법으로는 두 가지가 가능하다. 한 가지 방법은 반입되는 대상물과 빈 저장 위치 모두를 포함하는 것이고, 다른 방법은 단지 반입되는 대상물만 포함하는 것이다. 본 논문에서는 임시 염색체를 구성하는 첫 번째 방법을 이용하였다. 이 절에서는 두 가지 방법에 의한 유전자 알고리즘의 결과를 비교해 본다.

두 번째 방법으로 만들어진 반입되는 대상물만을 포함하는 임시 염색체에 교차변이를 수행하여 발생한 실행 불가능한 자식 염색체는 본 논문에서 제안한 교차변이 교정절차에 의해 교정될 수 없다. 이 경우에는 Nishimura *et al*. (2001)이 제안한 교차변이 교정절차로 실행 불가능한 자식 염색체에 대한 교정 작업이 가능하다. <Figure 10>는 제 3.4.1절에서 예제로 사용한 동일한 염색체에 수행한 교차변이 예제를 보여주고 있다.

두 가지 방법에 의한 유전자 알고리즘의 결과를 비교하기 위해서 본 논문은 두 번째 방법으로 만들어진 임시 염색체를 가지고 제 4.2절에서 기술한 유전자 알고리즘 실험을 반복하였다. 그리고 실험 결과를 <Figure 9>의 결과에 겹쳐 보았다. <Figure 11>은 두 가지 방법에 의한 평균 OFV 수렴 정도를 보여준다. <Figure 11>은 두 가지 방법에 의한 유전자 알고리즘의 결과에 차이가 거의 없음을 보여준다.

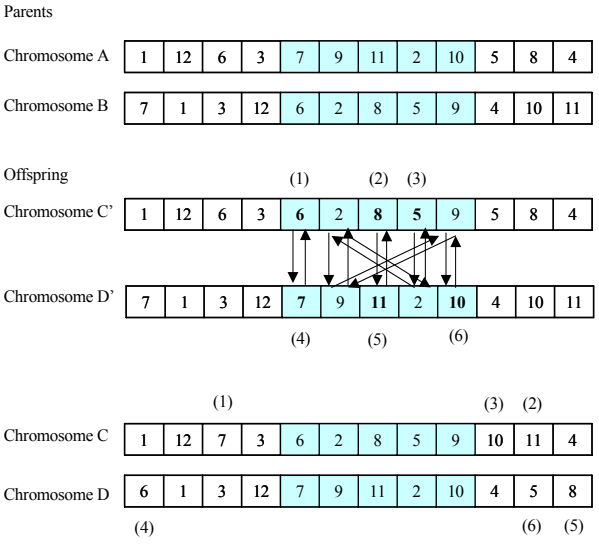


Figure 10. Example of another crossover processing

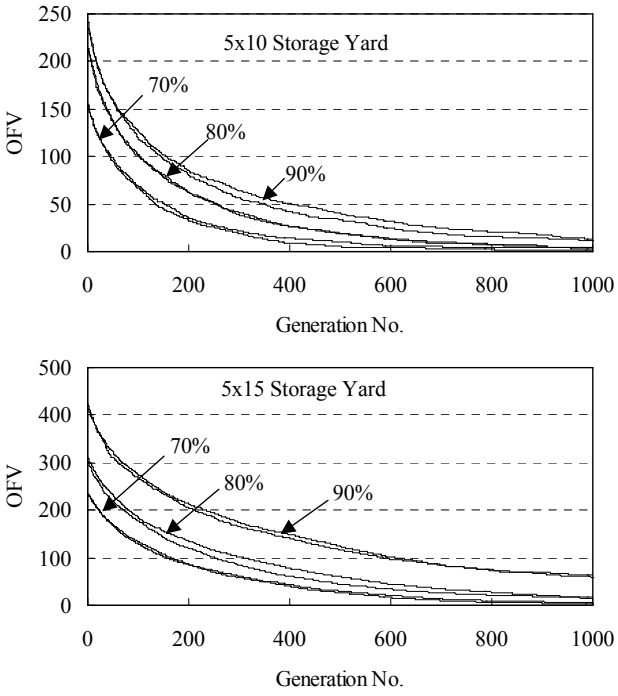


Figure 11. Comparison of the OFV convergences

5. 결 론

본 논문은 수리모형을 이용하여 해결하려는 시도를 아무도 하지 않은 평면적 저장 위치 할당 문제(PSLAP)를 소개하였다. PSLAP는 주로 조선소의 조립 블록 적치장 운영에서 발생한다. 항만 컨테이너 터미널에서 발생하는 저장 공간 및 위치 할당 문제와 달리, PSLAP 문제는 반입 및 반출되는 대상물의 이동이 X와 Y 방향으로만 엄격히 제한되어 있다.

본 논문은 PSLAP에 대한 수리모형을 수립하였으나, 이 문제는 NP-hard 문제에 속한다. 따라서 본 논문은 현실적인 규모의 큰 문제를 풀기 위해서 효율적인 유전자 알고리즘을 개발하였다. 그리고 실제 상황을 반영하는 시험 데이터를 이용하여 제안한 수리모형과 개발한 유전자 알고리즘의 성능을 검증하였다. 본 논문은 개발한 유전자 알고리즘을 실제적인 크기의 저장소에 적용 실험하였다. 다양한 실험을 통해 본 논문은

개발한 유전자 알고리즘이 PSLAP 문제를 해결하는데 좋은 대안이 될 수 있음을 보여 주었다.

참고문헌

Bazzazi, M., Safaei, N., and Javadian, N. (2009), A genetic algorithm to solve the storage space allocation problem in a container terminal, *Computers and Industrial Engineering*, **56**(1), 44-52.

Gen, M. and Cheng, R. (1997), *Genetic algorithms and engineering design*, New York : John Wiley and Sons.

Imai, A., Nishimura, E., and Papadimitriou, S. (2003), Berth allocation with service priority, *Transportation Research Part B*, **37**, 437-457.

Imai, A., Sasaki, K., Nishimura, E., and Papadimitriou, S. (2006), Multi-objective simultaneous stowage and load planning for a container ship with container rehandle in yard stacks, *European Journal of Operational Research*, **171**, 373-389.

Imai, A., Chen, H. C., Nishimura, E., and Papadimitriou, S. (2008), The simultaneous berth and quay crane allocation problem, *Transportation Research Part E*, **44**(5), 900-920.

Imai, A., Nishimura, E., Hattori, M., and Papadimitriou, S. (2007), Berth allocation at indented berths for mega-container ships, *European Journal of Operational Research*, **179**, 579-593.

Imai, A., Nishimura, E., and Papadimitriou, S. (2008), Berthing ships at a multi-user container terminal with a limited quay capacity, *Transportation Research Part E*, **44**, 136-151.

Kozan, E. and Preston, P. (1999), Genetic algorithms to schedule container transfers at multimodal terminals, *International Transactions in Operational Research*, **6**, 311-329.

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

Nishimura, E., Imai, A., and Papadimitriou, S. (2001), Berth allocation planning in the public berth system by genetic algorithms, *European Journal of Operational Research*, **131**, 282-292.

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

Sarker, R. and Newton, C. (2002), A genetic algorithm for solving economic lot size scheduling problem, *Computers and Industrial Engineering*, **42**, 189-198.

Vollmann, T. E., Berry, W. L., and Whybark, D. C. (1988), *Manufacturing planning and control systems*, Illinois : Irwin.

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.