IE Interfaces
Vol. 24, No. 1, pp. 78-86, March 2011.

조선소의 메가블록 조립작업장을 위한 공간계획알고리즘 개발

고시근[1†] · 장정희[2] · 최대원[3] · 우상복[3]

[1]부경대학교 시스템경영공학과 / [2](주)넥센타이어 / [3](주)삼성SDS SCM 컨설팅팀

Spatial Scheduling for Mega-block Assembly Yard in Shipbuilding Company

Shiegheun Koh[1] · Jeonghee Jang[2] · Daewon Choi[3] · Sangbok Woo[3]

[1]Dept of Systems Management and Engineering, Pukyong National University
[2]Nexen Tire Co., Ltd
[3]SCM Consulting Team, Samsung SDS Co., Ltd

To mitigate space restriction and to raise productivity, some shipbuilding companies use floating-docks on the sea instead of dry-docks on the land. In that case, a floating-crane that can lift very heavy objects (up to 3,600 tons) is used to handle the blocks which are the basic units in shipbuilding processes, and so, very large blocks (these are called the mega-blocks) can be used to build a ship. But, because these mega-blocks can be made only in the area near the floating-dock and beside the sea, the space is very important resource for the process. Therefore, our problem is to make an efficient spatial schedule for the mega-block assembly yard. First of all, we formulate this situation into a mathematical model and find optimal solution for a small problem using a commercial optimization software. But, the software could not give optimal solutions for practical sized problems in a reasonable time, and so we propose a GA-based heuristic algorithm. Through a numerical experiment, finally, we show that the spatial scheduling algorithm can provide a very good performance.

Keyword: shipbuilding, floating-dock, pre-erection, mega-block, spatial schedule

1. 서 론

조선산업에 있어 블록(Block)이란 선박 건조의 기본단위로서, 각 블록은 블록 조립작업장에서 조립된 후 도크(Dock)에서 전체 선박으로 만들어진다. <그림 1>에서 실선으로 나누어진 각 셀들이 하나의 블록이며, 그 크기는 대략 15m×15m, 중량은 100톤 내외이다. 선종 및 선형 그리고 공법에 따라 다르지만 대형 선박의 경우 하나의 선박을 150개 이상의 블록으로 구성하기도 한다.

블록을 선박의 형태로 구성해가는 작업장인 도크는 일반적으로 대부분의 조선소에서 병목자원으로 인식되고 있다. 따라서 조선 생산성을 제고하기 위해서는 선박이 도크에서 머무는 시간을 단축하여야 하며 이를 위해 각 조선소에서는 여러 가지 노력을 경주해왔다. 그 노력의 결과 중 가장 대표적인 것이

이 논문은 2008년도 정부(교육과학기술부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-2008-313-D01216).
†연락저자 : 고시근 교수, 608-737 부산광역시 남구 대연3동 599-1 부경대학교 시스템경영공학과,
Fax : 051-629-6478, E-mail : sgkoh@pknu.ac.kr
투고일(2010년 03월 22일), 심사일(1차 : 2010년 05월 19일, 2차 : 2010년 10월 18일), 게재확정일(2010년 10월 21일).

선행탑재(PE : Pre-erection)이다.

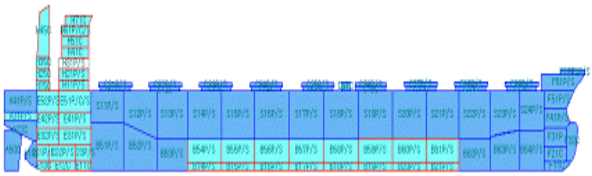


그림 1. 선박의 구조(Block Division)

선행탑재란 도크 내에서의 조립시간을 줄이기 위해 도크 옆에서 몇 개(2~3개 혹은 많은 경우 5~6개)의 블록을 미리 조립(Pre-erection)한 다음 골리앗 크레인으로 이 PE 블록을 도크 내부로 이동하여 조립 중인 선박에 탑재(Erection)하는 방식이다. 현재 이 방식은 국내 거의 모든 조선소에서 사용되고 있다.

하지만 골리앗 크레인의 처리가능 중량이 보통 800톤 이하이므로 PE 블록의 크기에는 제약이 있을 수밖에 없다. 이러한 문제를 해결하기 위해 일부 조선소에서는 도크를 육상에 두지 않고 <그림 2>와 같이 해상에 두기도 한다. 이것을 플로팅 도크(Floating-dock)라고 하며 보통 선박 1척을 그 안에서 건조한다. 이 플로팅 도크를 사용하는 경우에는 골리앗 크레인이 아닌 <그림 3>과 같은 형태의 해상크레인을 사용하여 블록을 탑재하게 되는데 해상크레인의 경우 처리가능 중량이 3,600톤에 이를 정도이므로 PE 블록의 크기제약이 크게 완화된다. 이렇게 해상크레인으로 처리하는 매우 큰 PE 블록을 Mega-PE 블록 혹은 단순히 메가블록(Mega-block)이라고 부른다. 실제 자료에 의하면, 해상크레인을 이용해 플로팅 도크에서 메가블록 조립방식으로 선박을 건조한 결과, 전체 블록의 수가 150여개에 이르는 선박을 건조하는데 단 10여 개의 메가블록이 사용되기도 하였다. 그 결과 도크 회전율이 급격히 높아져 전체적인 생산성이 크게 제고되었다.

그러나 도크에서의 작업시간이 줄어든 만큼 메가블록을 조립하는 작업장의 작업부하는 늘어나게 되는데 이 작업장은 공간적인 제약으로 인해 그 크기를 넓히기가 어렵다. 공간적인 제약이란 <그림 2>에서 볼 수 있듯이 해상크레인이 접근할 수 있는 바닷가에 위치해야할 뿐 아니라 해상크레인의 이동속도가 느리므로 플로팅 도크와 가까운 곳에 위치해야 한다는 것이다. 이러한 제약 하에서 늘어난 작업부하를 처리해야 하는 메가블록 작업장은 이제 조선소의 새로운 병목자원으로 인식되고 있으며, 따라서 조선 생산성을 제고하기 위해서는 메가블록 작업장의 효율적인 사용이 필수적이다. 특히 선박은 매우 고가이므로 납기지연 위약금이 매우 크고, 따라서 작업장의 효율적인 사용을 통해 작업일정을 단축하는 것은 다른 일반적인 제조업에 비해 파급효과가 훨씬 크다고 할 수 있다. 즉, 메가블록 조립작업장의 효율적인 사용계획을 통해 조선생산성을 제고하고 나아가 조선경쟁력을 강화할 수 있도록 하는 것이 본 연구의 궁극적인 목적이다.

그림 3. 해상 크레인

작업장의 효율을 높이기 위해서는 그 작업장의 병목자원을 효율적으로 사용해야 한다. 도크가 그렇듯이 메가블록 작업장의 경우에도, 생산되는 제품의 크기가 매우 크므로, 생산능력을 결정하는 가장 중요한 요인은 작업장의 면적이다. 결국 주어진 넓이의 작업장 안에 메가블록들을 가능하면 많이 배치하여 작업을 수행하는 것이 이 작업장의 생산성을 제고하는 가장 우선적인 수단이 되는 것이다. 따라서 본 연구에서는 주어진 계획기간에 납기(도크 탑재일)를 가진 메가블록들을 작업장 안에 가장 효율적으로 배치하는 공간계획(Spatial Scheduling) 방법을 찾고자 한다.

그런데 본 과제에서 다루는 메가블록 작업장은 해상 크레인을 이용해 블록을 운반한다는 특성으로 인해 블록을 들어 올리는 높이에 제약이 있어 작업장 안에 여유공간이 있다 하더라도 블록들을 병렬로 배치하는 것이 불가능하게 된다. 예를 들어 <그림 2>의 왼쪽에서 두 번째 및 세 번째 블록은 그 폭으로 볼 때 병렬로 배치하는 것이 가능하지만 해상 크레인의 이

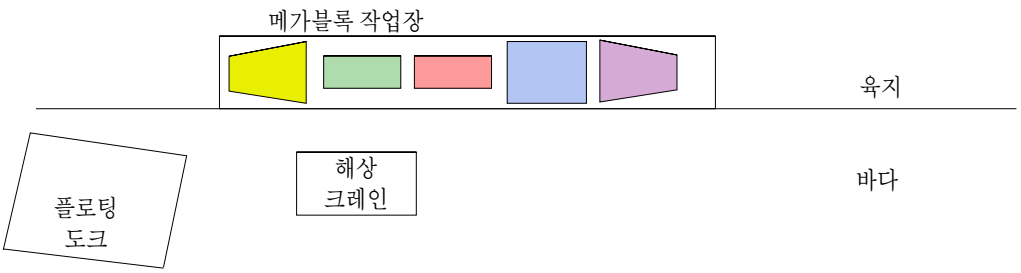


그림 2. 플로팅 도크와 메가블록 작업장

러한 작업특성에 의해 일렬로 배치된 것이다. 따라서 한 시점에서의 메가블록 작업장 배치문제는 블록의 형상에 관계없이 가로길이만을 고려하는 1차원 적인 문제가 되어 해결하기가 매우 쉬워진다. 그러나 본 연구의 대상문제는 한 시점이 아닌 일정한 기간 동안의 일정계획 문제이므로 시간축에 따른 변동성을 고려하여야 하며, 따라서 대상물의 길이만을 고려한 1차원 배치에 시간축을 더해 2차원 배치문제로 만들 수 있게 된다. 이러한 문제는 항만운영에서의 선박접안스케줄(Berth Scheduling) 문제와 매우 유사해진다. 선박이 항구에 들어오면 한정된 길이의 안벽(Quay)에 이미 정박 중인 다른 선박들 사이에 접안하여야 하는데, 이 경우에도 선박의 폭에 관계없이 선박의 길이가 접안 가능성을 결정하며 또한 각 선박의 정박일정이 모두 다르므로 시간축에 따른 변동성이 고려되어야 하기 때문이다.

선박접안스케줄링 문제는 최근 10여년에 걸쳐 연구가 이루어졌고 그 연구들은 기존의 Bin Packing 문제들(Lodi *et al*., 2002)을 응용하였다. 우선 Lim(1998)은 계획기간 내 필요한 안벽길이를 최소화하기 위한 접안스케줄 알고리즘을 개발하였다. 이 알고리즘은 선박이 항구에 도착하면 바로 접안하여야 하고, 일단 접안하면 작업이 끝날 때까지 이동이 불가능한 상황을 가정하였다. Park and Kim(2002)은 각 선박의 지연출항비용을 최소화하는 모형을 개발하고 Subgradient optimization 방법으로 해를 구하였다. 컨테이너 터미널을 대상으로 한 그들의 연구에서 지연출항의 발생원인은 먼저 접안한 선박으로 인한 접안불가능 뿐 아니라 정박위치에 따른 Quay 크레인 활용성도 포함한다. 이어서 Kim and Moon(2003)은 비슷한 문제를 Simulated annealing 방법으로 해결하였다. 그 후, Imai *et al*.(2005)은 각 선박의 정박시간이 정박위치에 따라 변하는 경우에 지연출항비용을 최소화하는 모형을 개발하고 그 모형에 대한 해를 구하기 위해 휴리스틱 방법을 제안하였다. 최근에는 Lee and Chen (2009)이 정박 중에 위치를 이동할 수 있는 상황에 대해 Neighborhood 탐색에 기반한 휴리스틱 알고리즘을 제안하였다.

본 연구에서 다루는 메가블록 작업장 문제는 기존의 선박접안스케줄링 문제와 유사한 점이 매우 많지만 문제의 제약조건이나 추구하는 목적의 관점에서 보면 다음과 같은 차이점을 갖고 있기도 하다.

1) 납기고정 및 공기단축 : 납기지연 페널티가 매우 비싼 선박의 특성으로 인해 도크에서의 탑재일은 반드시 지켜져야 하는 조건이다. 따라서 메가블록의 조립완료일(= 탑재일)은 고정되어 있다. 반면 잔업 등을 통해 조립공기는 단축할 수 있으며, 이것은 조립착수일을 늦추는 것이 가능하다는 의미이다.
2) 외주 가능성 : 대부분의 선박접안스케줄링 관련 연구에서는 접안일정을 유동적으로 처리하여 선박의 지연출항을 감안하였으나, 위에서 언급한 것처럼 메가블록 조립계획에서는 조립완료일을 고정되어 있어 배치가 불가능한 경우에는 외주로 처리하고 있다. 따라서 블록별로 자체생산 우선순위를 부여하여 이 우선순위가 높은 블록들을 우선적으로 배치한다.
3) 목적함수 : 선박접안스케줄링 문제의 목적함수들은 필요한 안벽길이의 최소화, 지연출항의 최소화 등이다. 본 연구에서는 주된 목적이 “외주의 최소화”이다. 즉, 계획기간 동안 가능하면 많은 블록들을 작업장 내에 배치하는 것이 제1의 목적이다. 또한 외주의 최소화와 더불어 작업비용을 최소화하기 위한 “공기단축의 최소화”도 필요하다.

이와 같은 차이를 반영하여 본 연구에서는 메가블록 조립계획을 위한 새로운 모형을 개발한다. 선박접안스케줄링 문제와 유사한 형태이지만 메가블록 조립작업장에서 요구되는 제약조건 및 목적함수를 고려한 새로운 모형을 만들고 그 타당성을 검증한다. 타당성이 검증된 모형에 대해서는 좀 더 빠른 시간에 좀 더 우수한 해를 찾아주는 알고리즘을 개발한다. 잘 알려진 것처럼 2차원 Bin Packing 문제는 NP-Complete 문제로서 (Lodi *et al*., 2002) 적절한 시간 안에 최적해를 구하기는 불가능하므로 유전 알고리즘에 기반을 둔 휴리스틱 알고리즘을 제시하고 그 성능을 평가한다.

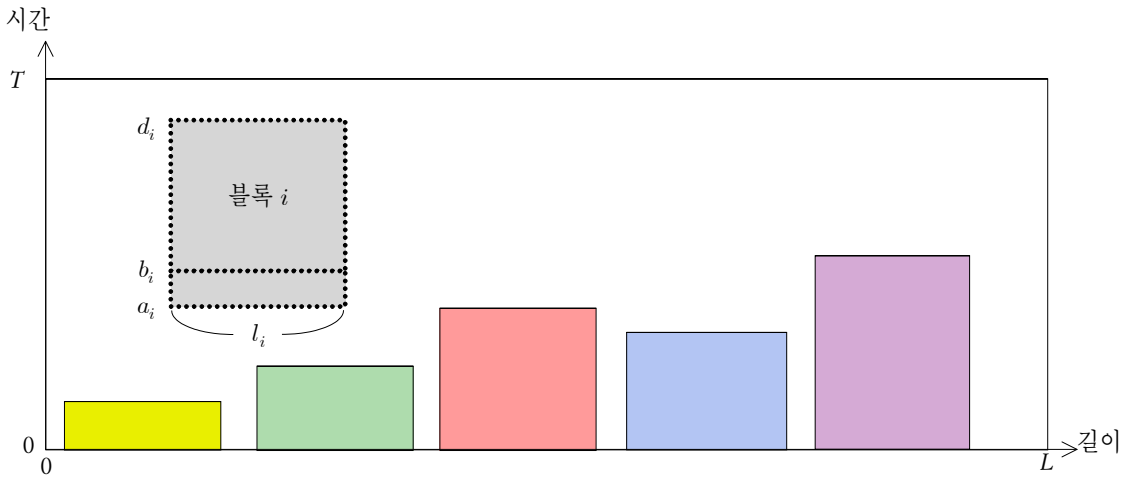
본 논문은 다섯 개의 장으로 구성되어 있다. 제 1장 서론에 이어 제 2장에서는 메가블록 조립작업장의 효율적인 공간계획을 위한 최적화 모형을 제시한다. 유전 알고리즘 기반의 휴리스틱 알고리즘 개발은 제 3장에서 다루며, 알고리즘의 성능평가를 위한 수치실험은 제 4장에서 다룬다. 마지막으로 제 5장에서는 논문에 대한 요약 및 향후 연구주제에 대해 논의하기로 한다.

2. 모 형

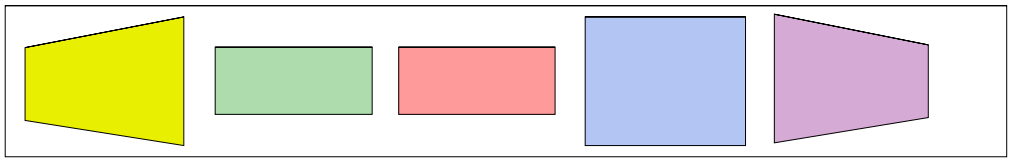
모형화를 위해 연구대상 문제를 <그림 4>와 같이 표현하기로 한다. 이 표현방식은 기존의 선박접안스케줄링 문제에서도 사용하였던 방식이다. 이 그림에서 실제 메가블록 조립작업장의 형태는 <그림 4>(b)와 같은데, 현재 이 그림이 보여주고 있는 것은 계획기간 시작시점의 작업장 상황이다. 작업장에 배치되어 있는 5개의 블록은 현 시점 이전에 배치되어 현재 작업 중인 블록이다. 이러한 상황은 <그림 4>(a)와 같이 길이-시간 2차원 평면에 표현될 수 있다. 이 그림에서의 직사각형은 가로가 블록의 가로 길이를, 세로가 블록의 조립공기를 표시한다. 단, 현재 작업 중인 블록의 경우에는 세로가 잔여공기를 의미한다.

계획대상 블록은 이 그림에서 점선으로 표시된 직사각형이다. 이 그림에서는 한 개이지만 실제로 계획대상블록은 여러 개 있으며 이 계획대상 직사각형들을 서로 겹치지 않게 배치하는 것이 본 연구의 주된 과제이다. 이 목적을 달성하기 위해 본 연구에서는 우선 수리적 최적화 모형을 개발한다.

수리적 모형의 개발을 위해 <그림 4>에 표시된 기호 및 앞으로 모형에서 사용할 기호들을 종류별로 설명하면 다음과 같다.



(a) 길이-시간 2차원 Bin Packing 문제 모형



(b) 계획시작시점의 실제 작업장 상황

그림 4. 메가블록 조립일정계획모형

<파라메터>

N = 계획기간 내에 나타나는 메가블록의 총 개수

n = 계획대상 메가블록의 개수(= N − 계획기간 초 작업 중인 블록 수)

L = 계획대상 메가블록조립 작업장의 길이

T = 계획기간

d_i = 블록 i의 완료시간

a_i = 블록 i의 착수가능시간(Earliest Start Time)

b_i = 블록 i의 최대 지연착수가능시간(Latest Start Time)

p_i = 블록 i의 자체생산 우선순위 점수(낮은 점수부터 외주)

l_i = 블록 i의 길이

<결정변수>

z_i = 블록 i를 자체생산하면(즉, 계획기간 내에 배치되면) 1, 그렇지 않으면 0

(x_i, y_i) = 블록 i가 배치된 경우 <그림 4>(a)의 좌표계에서 블록 좌하점의 좌표

<모형화에 필요한 부가변수>

z_{ij}^x = 블록 i와 블록 j가 모두 배치된 경우 블록 i가 블록 j의 좌측에 배치되면 1, 그렇지 않으면 0

z_{ij}^y = 블록 i와 블록 j가 모두 배치된 경우 블록 i가 블록 j의 아래에 배치되면(즉, 블록 i 완료 후 블록 j를 착수하면) 1, 그렇지 않으면 0

또한 대상문제를 명확하게 정의하기 위해 연구대상 시스템을 다음과 같이 가정한다.

(1) 계획대상 메가블록들은 착수가능시간(a_i)이 0이상이고 지연착수시간(b_i)이 T 이하이다.

(2) 일단 배치된 블록은 반출시까지 위치를 변경할 수 없다.

(3) 모든 블록들은 폭(세로 길이)이 작업장 폭보다 작고, 둘 이상의 블록들을 작업장의 폭 방향으로 병렬 배치할 수 없다. 따라서 배치가능성은 블록과 작업장의 가로 길이만 고려하면 된다.

(4) 블록은 작업장 내부에 배치되어야 한다. 즉, $0 \le x_i \le L - l_i$를 만족하여야 한다.

(5) 주어진 기간($a_i \le y_i \le b_i$)에 배치되지 않는 블록들은 외주를 준다. 각 블록은 여러 가지 특성에 의해 정해진 자체생산 우선순위 값(p_i)를 갖는데, 이 값이 작은 블록을 우선적으로 외주 준다.

(6) 각 블록의 반출시점(d_i)은 고정되어 있으나, 착수시점(y_i)은 착수가능시간(a_i)과 최대 지연착수가능시간(b_i) 사이에서 변동가능하다.

이러한 가정 아래 본 연구의 대상문제에 대한 수학적인 최적화 모형을 제시하면 아래와 같다.

$$\text{Maximize} \sum_{i=1}^{N} \{p_i z_i - (y_i - a_i) z_i\} \quad (1)$$

subject to

$$0 \le x_i \le L - l_i \text{ for all } i \quad (2)$$

$$a_i \le y_i \le b_i \text{ for all } i \quad (3)$$

$$x_i + l_i \le x_j + L(1 - z_{ij}^x) \text{ for all } i \text{ and } j,\ i \ne j \quad (4)$$

$$d_i \le y_j + L(1 - z_{ij}^y) \text{ for all } i \text{ and } j,\ i \ne j \quad (5)$$

$$z_{ij}^x + z_{ji}^x \le 1 \text{ for all } i \text{ and } j,\ i \ne j \quad (6)$$

$$z_{ij}^y + z_{ji}^y \le 1 \text{ for all } i \text{ and } j,\ i \ne j \quad (7)$$

$$z_{ij}^x + z_{ji}^x + z_{ij}^y + z_{ji}^y - z_i - z_j \ge -1 \text{ for all } i \text{ and } j,\ i \ne j \quad (8)$$

$$z_i : 0/1 \text{ integer for all } i \quad (9)$$

$$z_{ij}^x,\ z_{ij}^y : 0/1 \text{ integer for all } i \text{ and } j,\ i \ne j \quad (10)$$

모형에서 식 (1)은 자체생산의 최대화(특히 자체생산 우선순위가 높은 블록의 외주를 최소화) 및 공기단축의 최소화를 동시에 달성하기 위한 목적함수이다. 두 목적의 차원이 다르므로 이 목적함수의 결과값은 현실적인 의미를 갖지 못한다. 다만, 두 목적의 중요도에 따라 우선순위 값(p_i)의 크기를 조정함으로써 두 목적의 우선순위를 조정할 수 있다. 예를 들어, 현장에서는 일차적으로 외주를 최대한 줄이고 그 다음으로 공기단축을 최소화하는 것이 일반적인데, 그러한 경우에는 각 블록의 우선순위 값(p_i)을 최대 공기단축일수보다 충분히 크게 둠으로써 두 목적 사이의 우선순위를 조정하는 것이다. 식 (2)와 식 (3)은 가정 (4)와 가정 (6)을 표시한 것이다. 식 (4)는 x_i 및 x_j 값을 이용해 z_{ij}^x 의 값을 0 혹은 1로 만드는 제약이고 식 (5)는 y_i 및 y_j 값을 이용해 z_{ij}^y 값을 제어한다. 식 (6)과 식 (7)에 의하면 z_{ij}^x와 z_{ji}^x가 모두 1이거나 z_{ij}^y와 z_{ji}^y가 모두 1일 수 없다. 마지막으로 식 (8)에 의하면 z_{ij}^x, z_{ji}^x, z_{ij}^y, z_{ji}^y, $1-z_i$, $1-z_j$ 등 6개 값 중 적어도 하나는 반드시 1이 되어야 하며, 그 의미는 배치된 두 개의 블록은 서로 겹쳐질 수 없다는 것이다.

위 모형의 현실성을 검증하기 위해 다음과 같은 데이터를 사용한 예제를 풀어보기로 한다.

$N = 12$ (현재 작업 중인 블록 3개 포함), $L = 50$, $T = 35$,
d = (9, 3, 7, 14, 17, 21, 26, 30, 36, 44, 42, 43),
a = (0, 0, 0, 1, 2, 9, 9, 15, 19, 26, 24, 32),
b = (0, 0, 0, 3, 4, 12, 12, 18, 22, 29, 26, 34),
p = (10, 10, 10, 10, 20, 10, 20, 10, 20, 10, 20, 10),
l = (13, 9, 11, 12, 13, 11, 14, 14, 13, 12, 11, 15).

작업 중인 세 블록의 위치 및 일정을 표현하는 변수 값은 다음과 같다.

$x_1 = 0$, $x_2 = 18$, $x_3 = 39$, $y_1 = y_2 = y_3 = 0$,
and $z_1 = z_2 = z_3 = 1$.

최적화 문제를 해결해주는 상업용 소프트웨어(예를 들어 CPLEX 혹은 LINGO)를 사용하면 이 문제를 해결 할 수 있다. 본 연구에서는 LINGO 시스템을 사용해 최적해를 구하였으며 그 결과는 <그림 5>와 같이 표현할 수 있다. 이 그림에 의하면 한 개의 블록(10번 블록)이 배치되지 않았음을 알 수 있고 그 때의 목적함수 값을 계산하면 143(=150-7)이 된다.

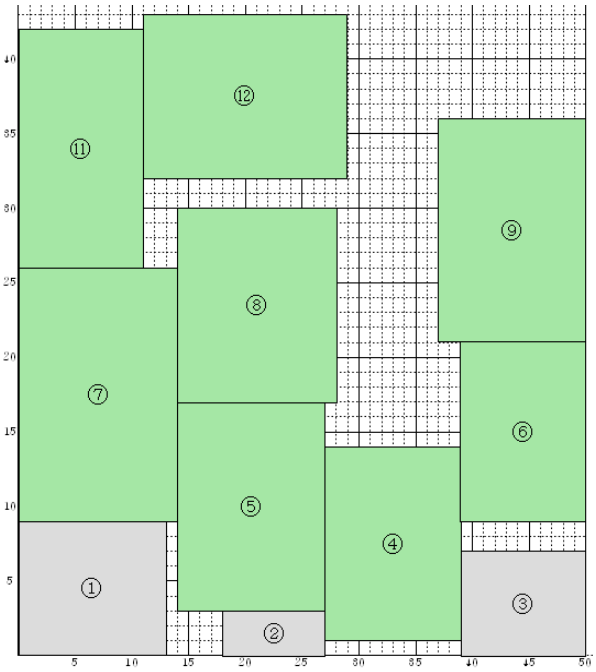


그림 5. LINGO에 의한 최적해

위 예제에서 볼 수 있듯이 규모가 매우 작은 문제의 경우 상업용 소프트웨어를 사용해 간단히 최적해를 구할 수 있다. 그러나 실제 상황에서는 문제의 크기(다시 말해 배치대상 블록의 개수)가 매우 커지게 된다. 실제로 본 연구의 동기를 제공한 기업의 경우 1개월에 처리하는 블록의 개수가 20~30개이고 계획대상기간은 최소 3개월에서 길게는 1년에 이르기 때문에 최적화 소프트웨어를 사용해 문제를 해결하는 것은 불가능하였다. 블록의 수가 15개인 예제의 경우에도 LINGO에서 실험하였을 때 24시간이 경과한 후까지 답을 구할 수 없어 프로그램을 강제로 종료할 수밖에 없었다. 따라서 본 연구에서는 적정한 시간 안에 최적해에 가까운 해를 구하기 위하여 유전알고리즘(GA; Genetic Algorithm) 기반의 휴리스틱 알고리즘을 설계하고자 한다.

3. GA 기반의 해법 개발

지난 30여년 간 매우 다양한 분야에서 사용되어온 유전알고리즘은 자연계의 적자생존 및 유전현상에 기초를 둔 확률적 탐색 알고리즘(Stochastic search algorithm)이다. 전통적인 탐색기법

과 다르게 GA는 임의로 생성된 초기해 집합으로부터 탐색이 시작된다. 이 해집단(Population)의 각 원소들은 염색체(Chromosome)라고 불리며 이 염색체가 실제 문제의 한 해에 해당된다. 이 염색체들은 세대(Generation)라고 불리는 반복(Iteration)이 진행됨에 따라 진화가 이루어진다. 각 세대에서는 모든 염색체들이 적합도에 대한 평가를 받게 된다.

3.1 염색체 표현 및 초기해 생성

해를 표현하는 방법을 결정하는 것은 유전알고리즘의 개발에 있어 가장 핵심적인 부분이다. 본 연구에서는 염색체를 이용해 계획대상 블록들의 배치순서를 표시하고자 한다. 순서를 표현하는 방법은 매우 많지만 본 연구에서는 0과 1사이의 랜덤수를 이용한 Random Keys 표현방법을 사용한다. 이 방법은 유전연산의 결과나 임의해 생성시 해의 Feasibility를 보장해주기 때문에 순서가 필요한 많은 문제(기계 스케줄링, 자원 배정, 차량경로 등)에 적용되어왔다. 계획대상 블록이 n개이므로 본 연구에서 사용하는 염색체는 n개의 유전자(즉, 0과 1사이의 랜덤 수)로 이루어진다. 각 염색체로부터 실제 블록배치계획을 생성(Decoding)하기 위한 세부적인 절차는 다음과 같다.

- Procedure *DECODE*

단계 1 : 각각의 염색체를 대상으로, n개의 유전자(각 유전자는 각 계획대상 블록과 대응됨)를 값이 증가하는 순서로 정렬한다. 정렬결과에 따라 대응되는 블록을 가능한 가장 지연된 시간에 배치(즉, $y_i = b_i$)한다. 여기서 가장 지연된 시간에 배치하는 이유는 앞에서 제시된 모형의 첫 번째 목적이 배치블록수를 최대화하는 것이기 때문이다. 일단 가장 지연된 시간에 배치함으로써 배치블록수를 최대화한 다음, 두 번째 목적(공기단축의 최소화)을 고려할 것이다.

단계 2 : 정렬된 순서에 따라 각 블록들을 다음과 같이 배치한다. 첫 번째 블록은 가능한 한 왼쪽에 붙여서 배치하고, 두 번째 블록은 가능한 한 오른쪽에 붙여서 배치하며, 세 번째 블록은 다시 가능한 한 왼쪽에 붙여서 배치하는 식으로 번갈아가며 좌우로 배치한다. 만약에 i번째 블록이 들어갈 자리가 없으면 $z_i = 0$로 놓는다.

단계 3 : 배치된 블록(즉, $z_i = 1$)에 대해서는 착수일(y_i)을 수정한다. 앞에서 모든 블록의 착수일을 가장 지연된 시간으로 두었기 때문에 두 번째 목적(공기단축의 최소화)을 달성하기 위해서는 각 블록의 착수일을 가능한 한 이른 시간으로 수정하여야 한다. 이 작업은 다른 블록들과 독립적으로 수행할 수 있기 때문에 쉽게 할 수 있다. 모든 블록에 대해 이 수정이 이루어지면 위에서 보았던 <그림 5>와 같은 형태의 계획결과를 얻을 수 있다.

앞서 보았던 예제를 사용해 이 절차를 설명한다. 블록의 총 개수는 12($N = 12$)이지만 3개는 이미 작업 중이므로 나머지 9개($n = 9$)의 블록에 대해 계획을 작성하면 된다. 따라서 9개의 랜덤수로 이루어진 염색체를 사용한다. 따라서 첫 번째 유전자는 <그림 5>의 4번째 블록과 대응된다. Decoding 대상 염색체가(0.35, 0.75, 0.12, 0.17, 0.87, 0.62, 0.99, 0.29, 0.15)라고 하자. 그러면 다음과 같은 절차를 통해 계획을 만들어낼 수 있다.

단계 1 : 염색체를 구성하는 9개의 유전자 값 크기에 따라 각 블록의 배치순서를 표시하면(5, 7, 1, 3, 8, 6, 9, 4, 2)와 같다. 즉, 3번째 유전자에 대응하는 블록(<그림 5>의 6번 블록)의 배치를 가장 먼저 고려한다.

단계 2 : 6번 블록에 대해 착수시점을 12로($y_6 = b_6 = 12$) 놓는다. 그러면 착수시점(12)에서 완료시점(21)까지 왼쪽 영역이 비어있으므로 이 블록은 왼쪽 끝 지점에 배치할 수 있다. 따라서 $x_6 = 0$ 및 $z_6 = 1$라고 놓을 수 있다. 두 번째 고려대상 블록은 9번째 유전자에 대응되는 12번 블록이다. 착수일자를 34 ($y_{12} = b_{12}$)로 하면 착수일(34)에서 완료일(43) 사이에 오른쪽 영역이 비어 있으므로 배치할 수 있고 그 결과는 $x_{12} = L - l_{12} = 35$ 및 $z_{12} = 1$이다. 다음 고려대상은 7번 블록이므로 왼쪽 영역에 배치가능성을 탐색하고, 그 다음은 11번 블록의 오른쪽 영역 배치가능성을 탐색하는 방식으로 진행된다. 이 단계의 결과는 <그림 6>에 나타나 있다. 이 그림을 보면 두 개의 블록(5번 및 9번)이 배치되지 않았음을 알 수 있다.

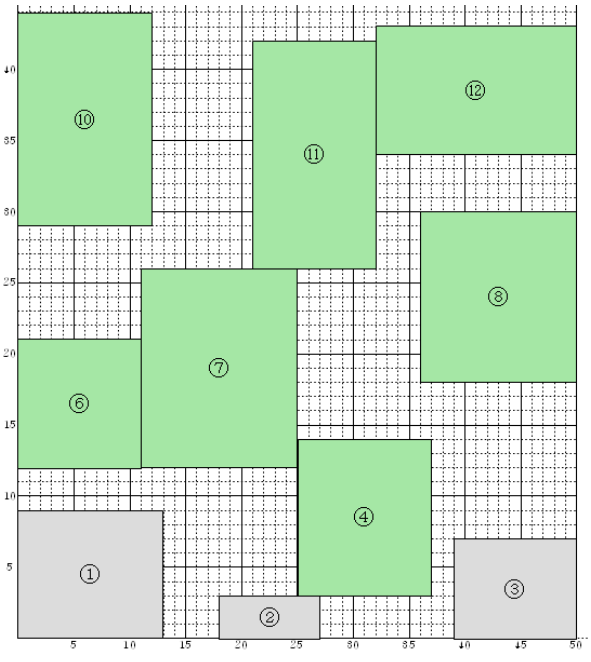


그림 6. 단계 2의 결과

단계 3 : <그림 6>을 보면 5개의 블록(6번, 7번, 8번, 10번, 12번)은 현재의 착수일보다 일찍 착수할 수 있음을 알 수 있다. 따라서 이 블록들의 정상 착수가능일(a_i)까지 착수일을 가능한 한도 내에서 당기면(그림에서는 블록의 밑변을 아래로 내리면) <그림 7>과 같은 결과를 얻을 수 있다.

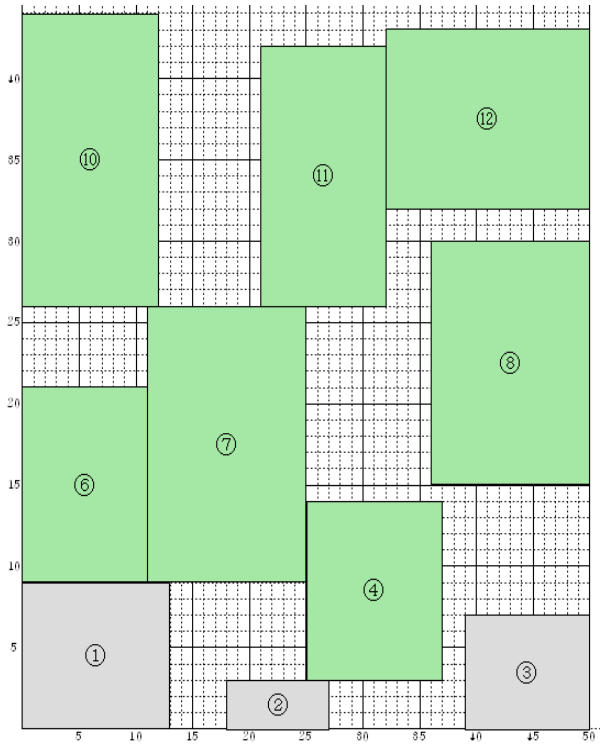


그림 7. 단계 3의 결과

3.2 목적함수와 적합도 함수

각 염색체에 대해 대응되는 계획이 하나씩 만들어지면 그 계획의 목적함수 값은 쉽게 계산할 수 있다. 위 예제의 경우 <그림 7>의 목적함수 값은 116이다. 이것은 배치된 블록들에 대한 우선순위 값의 합(120)에서 배치된 블록들의 공기단축일수 합(4)을 뺀 값이다. 본 연구의 대상문제는 최대화 문제이므로 이 목적함수 값을 그대로 적합도 함수값으로 사용해도 별 문제가 없다.

3.3 유전 연산

재생(Reproduction)은 각 염색체가 자신의 적합도 함수값에 따라 다음 세대로 복제되는 프로세스이다. 본 연구에서는 가장 널리 사용되는 복제방법인 룰렛 휠 방식을 사용한다.

교배(Crossover) 연산은 두 개의 염색체로부터 그들의 성질을 물려받은 두 개의 새로운 염색체를 만들어내는 과정이다. 본 연구에서는 가장 단순하고 전통적인 방식인 “One-cut exchange” 방식을 사용한다. 이 방식은 부모 염색체를 임의의 점에서 잘라서 좌우의 유전자를 서로 교환하는 방식이다.

돌연변이(Mutation) 연산에서도 가장 단순한 방법을 사용하였다. 주어진 확률에 의해 유전자를 임의로 선택하여 구간 [0, 1] 로부터 새로운 랜덤수를 생성하는 것이다.

또한 세대를 진행함에 있어 “Elitist strategy”를 적용하였다. 즉, 각 세대별로 가장 우수한 몇 개의(본 연구에서는 두 개) 염색체를 유전연산(교배 및 돌연변이) 없이 바로 다음 세대로 복제하는 것이다. 이렇게 함으로써 지금까지 발견된 가장 우수한 두 개의 해는 확실하게 다음 세대로 전달되는 것을 보장할 수 있게 된다.

4. 성능 평가

개발된 알고리즘의 성능을 평가하기 위해 소규모의 문제에 대해 LINGO를 사용한 최적해와 GA 결과를 비교해 보았다. 아래의 <표 1>은 LINGO로 얻은 최적값과의 백분율 오차를 분석한 것이다. 블록의 개수가 N = 8, N = 10, N = 12인 3가지 크기의 문제를 고려하였고 각각의 크기에 대해 10개의 문제를 랜덤하게 생성하여 각 문제 별로 10번씩 반복하여 실험한 결과이다. 교배확률과 돌연변이 확률은 여러 번의 실험을 거쳐 가장 우수한 결과를 제공해준 0.34와 0.04를 사용하였다. 오차는 (LINGO 최적값-GA 최적값)×100/LINGO 최적값 으로 계산하였고 그 값들에 대한 평균 오차와 최소 및 최대오차를 나타내었다.

결과를 보면 배치대상 블록의 수가 증가하여도 최적해와 비교해서 아주 좋은 결과를 보여주고 있다. 예외적인 경우도 있지만 대부분의 경우 평균 오차가 1% 내외이며 최소-최대 편차 또한 적어서 반복을 많이 하지 않아도 좋은 해를 찾을 수 있음을 알 수 있다. 특히 실험한 모든 경우에 최소 오차값이 0이었는데, 이것은 10회의 반복 중 적어도 한 번은 최적해를 찾았다는 것을 의미하므로, GA를 10회 정도 반복하고 그 중에서 가장 좋은 해를 선택한다면 상당히 좋은 해를 보장한다고 할 수 있다.

또한 <표 2>는 배치대상 블록의 수에 따른 결과 도출 시간을 초단위로 나타낸 것이다. 테스트 환경은 AMD Athlon(tm)64 X2 Dual Core Processor 4200+ CPU와 메모리 2기가 수준이었다. LINGO의 경우 고려대상 블록의 수가 늘어날수록 그 시간이 급격하게 늘어남을 알 수 있다. N = 12개인 경우 반복이 많아져서 최적해를 찾는데 상당히 많은 시간이 소요된다. 하지만 유전 알고리즘의 경우 블록의 수가 늘어남에도 소요시간은 크게 차이가 나지 않음을 알 수 있었다. 실제로 블록의 수가 13개 이상인 경우 LINGO로 해를 찾는 것은 매우 많은 시간이 필요하며 특히 15개 이상이 되는 경우에는 24시간 이하의 수행시간 내에 한 번도 해를 찾을 수 없었다. 반면 유전 알고리즘은 블록의 수가 증가하여도 비교적 짧은 시간에 좋은 해를 찾을 수 있었다.

표 1. 결과 분석

문제		블록 수		
		8	10	12
1	Mean	1.0	0.6	0.7
	Min	0.0	0.0	0.0
	Max	3.0	1.0	1.0
2	Mean	0.4	0.4	1.8
	Min	0.0	0.0	0.0
	Max	2.0	2.0	4.0
3	Mean	0.2	0.2	0.6
	Min	0.0	0.0	0.0
	Max	2.0	1.0	2.0
4	Mean	0.2	1.1	0.6
	Min	0.0	0.0	0.0
	Max	1.0	3.0	3.0
5	Mean	1.2	1.3	2.2
	Min	0.0	0.0	0.0
	Max	4.0	3.0	5.0
6	Mean	0.2	0.2	0.0
	Min	0.0	0.0	0.0
	Max	2.0	1.0	0.0
7	Mean	1.6	1.5	0.2
	Min	0.0	0.0	0.0
	Max	3.0	3.0	2.0
8	Mean	0.0	0.3	1.3
	Min	0.0	0.0	0.0
	Max	0.0	3.0	2.0
9	Mean	0.4	0.6	1.5
	Min	0.0	0.0	0.0
	Max	1.0	2.0	3.0
10	Mean	0.7	1.5	1.9
	Min	0.0	0.0	0.0
	Max	3.0	3.0	3.0

표 2. 평균 CPU-Time

	블록 수		
	8	10	12
LINGO	0.1	7.0	95.0
GA	0.8	1.3	2.5

5. 결 론

플로팅 도크에서 선박을 건조하는 경우 메가블록을 사용함으로써 도크의 회전율을 높이게 된다. 본 연구에서는 메가블록 조립작업장의 효율적인 사용을 위한 모형을 개발하였다. 선박 접안스케줄링 문제에 기초를 두고 메가블록 조립작업장에서 요구되는 제약조건 및 목적함수를 고려한 새로운 모형을 만들고 상업용 최적화 소프트웨어를 사용해 그 타당성을 검증하였다. 현실적인 크기의 문제에 대해서는 상업용 소프트웨어를 사용한 최적해를 구하는 것이 불가능하므로 현실적인 시간에 우수한 해를 찾아주는 유전알고리즘 기반의 해법을 개발하였다. 수치실험을 통해 테스트한 결과 개발된 해법은 빠른 시간 안에 매우 우수한 결과를 보여 주었다.

본 연구의 결과는 다음과 같은 상황으로 확장되는 것이 필요하다. 본 연구에서는 작업 시작에서 작업 종료까지 블록의 크기가 일정하다고 가정하였으나 실제로 블록은 조립작업이 진행됨에 따라 그 크기가 커지게 된다. 이러한 상황을 고려하면 <그림 4>에서 보았던 4각형은 밑변에 비해 윗변이 긴 새로운 형태의 도형이 되어 새로운 해결방안을 모색하여야 할 것이다. 현재 저자들은 이러한 문제에 대한 해법을 연구하는 중이다.

참고문헌

Imai, A., Sun, X., Nishimura, E., and Papadimitriou, S. (2005), Berth allocation in a container port : using a continuous location space approach, *Transportation Research Part B*, **39**, 199-221.

Kim, K. H. and Moon, K. C. (2003), Berth scheduling by simulated annealing, *Transportation Research Part B*, **37**, 541-560.

Lee, Y. and Chen, C.-Y. (2009), An optimization heuristic for the berth scheduling problem, *European Journal of Operational Research*, **196**, 500-508.

Lim, A. (1998), The berth planning problem, *Operations Research Letters*, **22**, 105-110.

Lodi, A., Martello, S., and Monaci, M. (2002), Two-dimensional packing problems: A survey, *European Journal of Operational Research*, **141**, 241-252.

Park, K. T. and Kim, K. H. (2002), Berth scheduling for container terminals by using a sub-gradient optimization technique, *Journal of the Operational Research Society*, **53**, 1054-1062.

고 시 근
고려대학교 산업공학과 학사(1986)
한국과학기술원(KAIST) 산업공학과
석사(1988) 및 박사(1993)
현재 : 부경대학교 시스템경영공학과 교수
관심분야 : 생산경영, 생산정보시스템, SCM

장 정 희
부경대학교 시스템경영공학과 학사(2007) 및
석사(2009)
현재 : 넥센타이어 연구개발본부 연구기획팀
연구원
관심분야 : 생산관리, 스케줄링, 컴퓨터 응용

최 대 원
한국과학기술원(KAIST) 산업공학과
학사(1999), 석사(2001), 박사(2004)
현재 : 삼성SDS SCM 컨설팅팀 수석컨설턴트
관심분야 : 생산관리, 스케줄링, 확률모형

우 상 복
서울대학교 산업공학과 학사(1989),
석사(1991), 박사(1997)
현재 : 삼성SDS SCM 컨설팅팀 수석컨설턴트
관심분야 : SCM, 스케줄링, 시뮬레이션