한국항해항만학회지 제38권 제3호
J. Navig. Port Res. Vol. 38, No. 3 : 299-305, June 2014 (ISSN:1598-5725(Print)/ISSN:2093-8470(Online))
DOI : http://dx.doi.org/10.5394/KINPR.2014.38.3.299

트랜스포터 결합을 고려한 조선소 블록 운반 일정계획

신재영* · † 박나현

* 한국해양대학교 물류시스템공학과 교수, † 한국해양대학교 대학원

Transporter Scheduling with Transporter Combination in Shipbuilding

Jae-Young Shin · † Na-Hyun Bak*

** Department of Logistics Engineering, National Korea Maritime University, Busan 606-791, Korea*
† Graduate school of National Korea Maritime University, Busan 606-791, Korea

요 약 : 트랜스포터란 조선소에서 작업장 간에 조선 블록을 운반하는 장비이다. 트랜스포터 일정 계획을 효율적으로 하는 것은 선박 건조 과정에서 매우 중요한 역할을 한다. 블록 운반에 있어서의 트랜스포터 일정 계획에 관한 몇몇 연구들이 수행되었다. 이러한 연구들은 트랜스포터 결합을 고려하지 않았다. 본 논문에서는 결합을 고려한 효율적인 트랜스포터 일정 계획 모형을 제안한다. 수리적 모형에서는 운영 비용을 최소화하면서 트랜스포터 간의 작업 부하를 가능한 균등하게 하는 것을 목적으로 하였다. 또한 본 모형의 해를 탐색하기 위해 타부 서치 기반의 세 가지 휴리스틱 알고리즘을 제시하였으며 컴퓨터 실험을 통해 제안된 휴리스틱들의 효율성을 검증하였다.

핵심용어 : 트랜스포터 일정 계획, 조선소, 차량 경로 문제, 타부 서치, 휴리스틱 알고리즘

Abstract : *In a ship-building, a transporter is the moving equipment to transport ship blocks from a workshop to another in a shipyard. The efficient scheduling of transporters has an important role for the operation of a building ship to be completed on schedule. There are the previous studies on the transporter scheduling for moving blocks in a shipyard. These studies have no consideration for the transporter combination to increase the productivity of moving blocks. This paper presents an efficient transporter scheduling model considering transporter combination explicitly. The objective of this model is to minimize the operational cost and maintain the workload balance among transporters. we also present three heuristics algorithms based on tabu search for finding the solution of the model. The efficiency of the proposed heuristics are verified with several computational tests.*

Key words : *Transporter scheduling, Shipbuilding, Vehicle Routing Problem, Tabu search, Heuristic algorithm*

1. 서 론

조선소에서 블록은 선박 건조의 기본적인 단위로서 재공품 형태로 존재하며 도크에 탑재되기 전까지 조선소 내의 여러 곳에 분산되어 관리된다. 선박의 특성에 따라 생산되는 블록의 크기 및 개수가 다양하며 대형 선박의 경우 약 200개 이상의 블록으로 구성되고 대형 조선소의 경우 하루 약 300개 이상의 블록이 이동한다.

블록마다 정해진 일정 계획이 있으며 일반적으로 이전 공정이 완료되면 적치장으로 운반하여 보관되었다가 일정 계획에 따라 다음 공정으로 운반된다. 블록은 도크에서 탑재되기 전까지 이와 같은 이동과 적치를 반복하게 된다. 다음의 Fig 1은 선박 건조 공정의 흐름 중에 트랜스포터를 이용한 블록의 이동이 필요한 구간을 나타낸다.

국내 조선업계들이 해양플랜트까지 사업영역을 확장함에 따라 조선소 내에서 플랜트와 선박이 함께 건조되고 있다. 이에 따라 관리해야 할 블록의 수량도 증가하여 적절한 블록 일정 계획을 세우고 이를 정확히 수행하는 것이 더욱 중요해졌다. 만일 계획과 다른 시기에 블록 운반이 이루어지게 되면 전후 작업 공정의 일정에 영향을 줄 수 있으며 이는 전체 일정의 지연을 초래할 수 있다. 선박은 고가이므로 위약금이 커 다른 제조업에 비해 작업 일정을 준수하는 것이 더욱 중요하게 여겨진다. 게다가 납기일이 다른 여러 선박들이 동시에 건조되므로 전체 일정에 영향을 주지 않게 각각의 일정계획을 준수하는 것이 중요하다. 그러므로 적절한 블록 운반 계획을 세우고 이행하는 것은 물류비 감소로 인해 매출 대비 순이익 증가를 가져올 뿐만 아니라 납기일을 준수할 수 있게 됨으로써 신뢰도 및 경쟁력을 향상시킬 것으로 예상한다.

트랜스포터(Transporter)란 블록을 이동시키는 유압식 특수 운반 장비이며 수 백톤에서 1,000톤에 이르기까지 적재 중량에 따라 여러 종류로 존재한다. 트랜스포터는 자체 동력 장비(엔진)를 탑재하고 있는 독립적 장비이며 다른 트랜스포터

* 대표저자 : 종신회원, shinjy@hhu.ac.kr 051)410-4335
† Corresponding author : 연회원, pnh891208@naver.com 051)410-4931

또는 모듈 트랜스포터와 연결하게 되면 적재 중량을 늘일 수 있다. 즉, 500톤 블록을 이동시키기 위해 200, 300톤의 트랜스포터를 연결하거나 200톤의 트랜스포터 3대를 연결하여 운반할 수 있다. 트랜스포터는 장비 구입비 및 유지보수비가 고가인 장비이므로 효율적으로 운영하는 것이 물류비 절감에 있어 중요하다. 블록 크기에 맞춰 종류별로 트랜스포터를 보유하는 것보다 적절한 용량으로 구비하여 상황에 맞게 결합한다면 적재율을 높여 물류비용을 절감할 수 있다.

최근 조선소에서 드라이도크의 수요를 충족시키기 위한 대안으로 플로팅 도크 시스템을 도입하고 있다(Kim et al, 2008). 플로팅 도크로 PE 블록을 운반하는 장비로 해상크레인, 모듈 트랜스포터 등이 있으며 트랜스포터를 결합하여 대형블록을 운반하는 것이 해상크레인을 이용하는 것보다 안전하며 더 큰 블록을 운반할 수 있다는 점에서 경쟁력이 있다. 동적인 조선소 환경에서 트랜스포터를 결합하여 블록을 운반하게 되면 다양한 운반 계획이 가능하고 유연하게 대처할 수 있다는 점에서 본 연구에 의의가 있다.

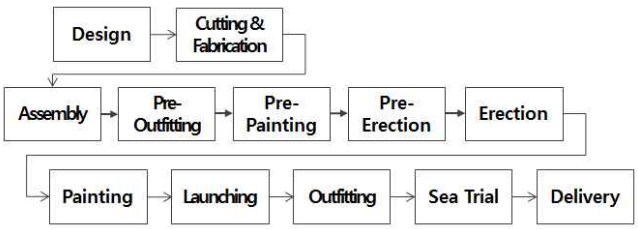


Fig 1 Flow Chart for Shipbuilding Process

이렇듯 조선 물류의 원활한 흐름에 있어 중요한 장비임에도 불구하고 트랜스포터 일정 계획에 관한 연구는 많이 이루어지지 않았다. Yim et al(2008)은 공주행을 최소로 하는 트랜스포터의 운반 계획 알고리즘을 제안하였다. 운반 거리를 결정하기 위해 Dijkstra 알고리즘을 이용하였고, 최적의 블록 할당 및 운반순서를 결정하기 위해 개미 알고리즘과 유전 알고리즘을 혼합한 혼합형 알고리즘으로 문제를 해결하였다. Lee et al(2008)은 조선소에서 계획기간동안 갑작스런 블록 수송정보의 변경이나 트랜스포터의 고장 등이 발생하게 될 경우, 동적인 환경에서 트랜스포터 일정 계획을 위해 4가지 휴리스틱들을 제안하고 각각의 성능을 비교하였다. Lee et al(2009)은 같은 동적인 환경에서 네트워크 흐름 기반의 할당 휴리스틱 알고리즘(NFA)을 제안하여 이전 논문에서 가장 효율이 좋았던 휴리스틱과 비교하였다. Park et al(2012)는 트랜스포터 간의 작업 부하를 균등하게 하기 위해 GRASP(greedy randomized adaptive search procedure) 알고리즘을 제안하였다. 위의 선행 연구들에서는 하나의 블록에 하나의 트랜스포터를 할당하는 문제를 고려하였으며, 트랜스포터의 차량 결합을 고려한 연구는 거의 수행되고 있지 않음을 알 수 있다.

이외에 트랜스포터 일정 계획 문제와 유사한 문제로 컨테이너 터미널에서의 야드 트랙터 운영을 들 수 있다(Shin et al, 2009; Chung et al, 2012). 그러나 야드 트랙터는 차량 크기와 운반해야 할 컨테이너가 정형화되어있으므로 본 문제와 차이가 있다.

본 연구에서는 트랜스포터 결합을 고려하여 정해진 작업 일정을 지키고 물류비를 줄이기 위해 트랜스포터 보유량을 최소화하고 할당된 트랜스포터들의 운행 시간을 평준화하는 수리적 모형을 구축한다. 최소화에만 초점을 둘 경우 특정 차량에 작업이 몰리게 될 수 있으며 예상 시각에 맞춰 작업을 완료하지 못할 경우에 문제가 발생할 수 있다. 이와 같은 위험을 줄이기 위해 운행 시간의 평준화가 필요하다. 목적함수에서 차량 대수 최소화를 고려하는 이유는 얼마나 효율적으로 현재 차량을 운영 중인지 알 수 있으며 설계적인 측면에서 앞으로 필요한 차량 대수를 산정할 수 있다. 또한 유휴 차량을 확보하여 차량 계획과 작업에서 차이가 생길 경우 빠른 대처가 가능하다. 만일 정해진 차량으로 운영하는 것에만 목적이 있을 경우 목적함수에서 차량최소화에 관한 가중치를 0으로 두면 된다. 위의 내용을 바탕으로 초기해로 사용될 휴리스틱 알고리즘을 제안하고 메타 휴리스틱 중 하나인 타부서치 알고리즘을 이용한 총 3가지의 휴리스틱 알고리즘을 제시하여 각각의 결과를 비교해 보고자 한다.

2. 수리적 모형

본 연구에서 풀고자 하는 문제는 여러 대의 트랜스포터 결합으로 하나의 블록을 운반하는 것이므로 이는 하나의 블록을 쪼개어 여러 대에 실을 수 있는 상황에서의 차량 경로 문제인 SDVRP(Split Deliveries Vehicle Routing Problem)와 유사하다고 볼 수 있다.(Dror and Trudeau, 1990; Dror et al.,1994) 그리고 전후 작업 일정에 영향을 주지 않기 위해 각각의 블록은 정해진 기간 내에 운송이 되어야한다는 점에서 시간창이 존재하는 SDVRP상황인 VRPTWSD (The vehicle routing problem with time windows and split deliveries)가 본 논문에서 다루고자 하는 모형과 가장 유사하다고 볼 수 있다.(Ho and Haugland, 2004; Desaulniers, 2010) 하지만 시간창 외에도 Pick-up and Delivery와 다종(heterogeneous)의 차량타입을 고려한다는 점에서 이전 연구들과 차별화된다. 또한 모든 차량이 도착해야 운반을 시작할 수 있고 실제로 짐이 나누어지지 않는다는 점에서 SDVRP와는 다른 성격을 지닌다.

블록 이동을 위한 트랜스포터 할당 문제에 대해 정의 및 가정 사항은 다음과 같다.

- 각 트랜스포터별로 가용 적재량이 다르고, 보유 비용이 다르다.
- 운반 블록은 트랜스포터에 의해 한 번 운반된다.
- 블록을 한 대의 트랜스포터 단독으로 운반하거나 두 대 이상의 트랜스포터들을 결합하여 운반할 수 있다.
- 트랜스포터를 결합하여 운반하도록 계획된 경우, 하나의

블록에 계획된 모든 트랜스포터가 도착할 때까지 다른 트랜스포터들은 대기한다. 즉, 모든 트랜스포터가 준비가 되었을 때 실제로 운송이 가능하다.

· 각 블록의 운송해야 할 시점에 대한 기간은 정해져 있으며 그 기간 내에 운송이 착수되어야 한다.
· 각 블록의 크기 및 운송 시간은 다르다.
· 다른 두 블록간의 운송 시간 및 이동 시간은 주어져 있다.
· 트랜스포터는 2개 이상의 블록을 실은 채로 이동할 수 없다. 즉, 블록 pick-up이 발생하면 해당 블록이 delivery 될 때까지 다른 블록을 pick-up할 수 없다.

모형에서 사용될 Parameter 및 입력 변수는 다음과 같다.

· r_i : 블록 i가 운반 가능한 시점.
· s_i : 전후 작업에 지장을 주기 않게 하기 위한 블록 i 출발 시각. (일정 계획상 운반 완료되어야 하는 시각 - 블록 i 의 운반시간)
· w_i : 블록 i 의 중량.
· c_k : 트랜스포터 k 의 적재 능력.
· m_{ij} : 블록 i 하역 지점에서부터 블록 j 적재 지점까지 이동하는데 걸리는 시간.
· t_i : 블록 i 가 실제 pick-up되는 시각.
· y_k : 트랜스포터 k 가 당일 작업에 투입되면 1, 아니면 0.
· x_{ij}^k : 트랜스포터 k 가 블록 i 를 운반하고 다음 순서로 블록 j 를 운반하면 1, 아니면 0.
· B_k : 트랜스포터 k 의 총 운행 시간.
· V : 트랜스포터 k 의 집합. $V = \{1, \cdots, m\}$
· N : 블록 i 의 집합. $N = \{1, \cdots, n\}$

여기서 가상의 출발, 도착 데포를 0으로 가정한다. Fig 2는 본 연구에서 정의한 블록 운송의 흐름 예제이다. 예를 들어 트랜스포터 T1과 T2가 있다고 했을 때 T1의 적재 중량은 200톤, T2의 적재 중량은 300톤이고 블록 2의 중량이 500톤이라고 했을 때 T1, T2 각각으로 운반할 수 없지만 두 차량을 결합한다면 블록 2를 운반할 수 있다. 또한 m_{35}는 블록 3을 p_3에서 pick-up하여 d_3에 delivery한 다음 p_5로 가서 블록 5를 운반하기까지 걸리는 시간이다.

앞의 가정 및 제약조건을 바탕으로 수립한 수리적 모형은 다음과 같다.

$$Min\, z = \alpha \sum_{k \in V} \gamma_k y_k + \beta \max_{k \in V}\{B_k\} \qquad (1)$$

subject to

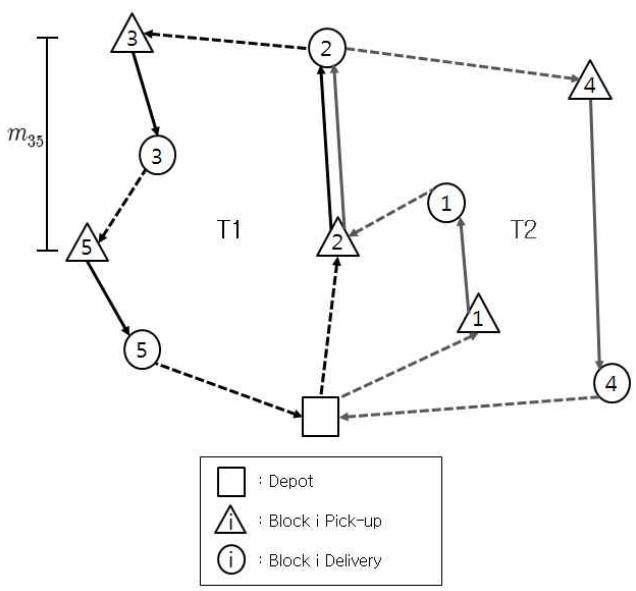


Fig 2 Block transport

$$B_k = \sum_{i \in N} \sum_{j \in N} m_{ij} x_{ij}^k \qquad k \in V \qquad (2)$$

$$y_k = 0 \rightarrow \sum_{i \in N} \sum_{j \in N} x_{ij}^k = 0 \qquad k \in V; i \neq j \qquad (3)$$

$$\sum_{k \in V} \sum_{j \in N} c_k x_{ij}^k \geq w_i \qquad i \in N; i \neq j \qquad (4)$$

$$\sum_{i \in N} \sum_{k \in V} x_{ij}^k \geq 1 \qquad j \in N; i \neq j \qquad (5)$$

$$\sum_{j \in N} x_{0j}^k = y_k \qquad k \in V \qquad (6)$$

$$\sum_{i \in N} x_{ip}^k - \sum_{j \in N} x_{pj}^k = 0 \qquad p \in N; k \in V; i \neq p; j \neq p \qquad (7)$$

$$x_{ij}^k = 1 \rightarrow t_i + m_{ij} \leq t_j \qquad k \in V; i \in N; j \in N; i \neq j \qquad (8)$$

$$r_i \leq t_i \qquad i \in N \qquad (9)$$

$$t_i \leq s_i \qquad i \in N \qquad (10)$$

$$y_k \in \{0, 1\} \qquad k \in V \qquad (11)$$

$$x_{ij}^k \geq 0 \qquad k \in V; i \in N; j \in N \qquad (12)$$

모형에서 식 (1) 은 조선소에서 이용하는 트랜스포터 차량 대수의 최소화 및 각 차량의 운영시간의 평준화를 목적으로 한다. γ_k는 차량 보유 비용에 따라 달라지고, α, β는 어떤 목적에 비중을 더 많이 두느냐에 따라 조절할 수 있는 가중치이다. 식 (2) 는 트랜스포터 k의 운행 시간을 나타내고, 식 (3) 는 트랜스포터 차량 k 의 당일 가동에 관한 제약식이다. 식 (4) 는 차량 크기에 따라 실을 수 있는 블록 크기의 제약을 나타내고, 여러 대로 운반이 가능한 가정이므로 운반하는 차량 적재량들의 합과 블록 크기를 비교한다. 식 (5) 은 각 블록을

한 번은 방문해야한다는 제약식이며, 식 (6) 은 당일 사용되는 차량은 노드 0에서 출발해야한다는 제약식이다. 식 (7) 은 네트워크 흐름보존법칙을 나타낸다. 식 (8) 은 트랜스포터 여러 대로 블록 j 를 운반하는 경우에 할당된 모든 차량이 준비가 될 때 운반이 가능하다. 즉, 할당된 차량 중 가장 늦게 준비가 되는 차량의 시각으로 다른 차량들의 운반 시각을 맞추는 제약이다.

3. 해법

일반적으로 VRPSDTW는 NP-Hard로 알려져 있으며 문제의 크기가 커질수록 계산시간이 기하급수적으로 증가한다 (Dror et al, 1990). 하지만 조선소에서는 작업자의 숙련도에 의해 작업 속도가 결정되는 작업이 많고 조선 자체가 프로젝트의 성격을 띠므로 건조 중 설계 변경이 빈번하다. 또한 트랜스포터의 고장 및 정비로 인해 계획의 변경이 불가피할 경우가 종종 발생하므로 본 연구에서는 동적 환경에서도 적용할 수 있도록 빠른 시간 내에 근사 최적해를 낼 수 있는 휴리스틱 알고리즘을 제안한다. 제안하는 초기해 생성을 위한 경로 생성 알고리즘과 타부서치 알고리즘의 상세 단계는 다음과 같다.

3.1 경로 생성 휴리스틱 알고리즘

Step 1 블록 및 트럭을 정렬. 현재 대기 중인 트랜스포터의 준비 시각(r^k)을 0으로, 운행 중인 트랜스 포터의 준비 시각을 해당 작업이 끝나는 시각으로 초기화. $i = 1$

Step 2 $s_i < r^k$ 를 제외하고 이미 할당되어 있는 트랜스포터($\sum_i \sum_j x_{ij}^k > 0$) 중에서 트랜스포터의 운행 시간($\sum_i \sum_j m_{ij} x_{ij}^k$) 이 가장 작은 트랜스포터 k' 선택.

$$k' = \arg\min_{k \in V} \left\{ \max\left(\sum_i \sum_j m_{ij} x_{ij}^k , 0 \right) \right\}$$

위에 해당되는 k가 없다면 할당되지 않은 트랜스포터들 ($\sum_i \sum_j x_{ij}^k = 0$) 중 가장 비용이 적게 발생하는 k' 선택.

Step3 if ($c_k < w_j$) Step2를 반복. $i = i + 1$
else, Step4를 진행.

Step4 if ($i \le n$) Step 2를 진행한다.
else, 종료.

3.2 타부서치 알고리즘

Step 1 (Initialize)
s_0 = 경로 생성 휴리스틱 알고리즘에서 얻은 해.
TabuList = Φ

Step 2 (Neighborhood Search)
임의의 트랜스포터 k'에 할당된 블록 중 임의로 하나의 블록 i를 선택.
이웃해 탐색 방법에 의해 이웃해 집단($N(s)$)을 생성.

Step 3 (Evaluation and Update)
평가 함수 값은 수리적 모형에서 수립한 목적함수 값과 동일하게 정의. 이웃해 중 평가 함수 값이 가장 좋은 해를 선택.
($s' = \arg\min_{a \in N(s)} \{f(a)\}$)
if ($f(s) > f(s')$) then $s \leftarrow s'$ TabuList.Add(i)
if (!Stopping Rule) then Step 2 반복

Step 4 (End)
best known solution = s 로 종료

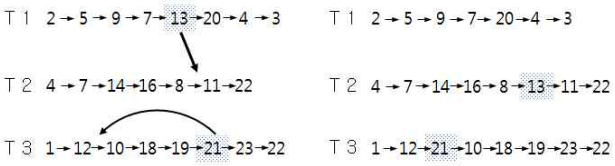
본 연구에서 제안한 타부 서치 알고리즘에서 이웃해 탐색에 쓰이는 방법은 3가지이며 각 방법에 대한 설명과 구현을 위한 과정을 다음과 같이 서술한다.

1. Relocate operator(Fig 3)

Step 2의 트랜스포터 k'에서 블록 i 를 제거

$$w' = w_i - \sum_{k \in V} \sum_{j \in N} c_k x_{ij}^k - c_{k'}$$

for $k_1 = 1, \cdots, m$ do
if ($w' \le c_{k_1}$ and $\sum_{j \in N} x_{ij}^k = 0$) then
Repeat (k_1에 할당된 모든 블록 j에 대해서)
if ($j \notin N_i^1$ and $j \notin N_i^2$) then
블록 j의 전 또는 후에 블록 i를 배치하여,
feasible(제약을 모두 만족)한 해를 이웃해에 추가



(a) Initial (b) Neighborhood
Fig 3 Relocate operator

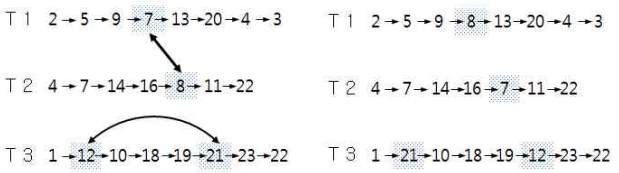
2. Swap operator(Fig 4)

$$w' = w_i - \sum_{k \in V} \sum_{j \in N} c_k x_{ij}^k - c_{k'}$$

for $k_1 = 1, \cdots, m$ do
if ($k_1 == k'$) or ($w' \le c_{k_1}$ and $\sum_{j \in N} x_{ij}^k = 0$) then
Repeat (k_1에 할당된 모든 블록 j에 대해서)

if ($j \notin N_i^1$ and $j \notin N_i^2$) then

블록 j와 블록 i의 순서를 교체, feasible 해를 이웃해에 추가



(a) Initial (b) Neighborhood

Fig 4 Swap operator

3. 2-Opt operator(Fig 5)

$$w' = w_i - \sum_{k \in V} \sum_{j \in N} c_k x_{ij}^k - c_{k'}$$

for $k_1 = 1, \cdots, m$ do

if ($k_1 == k'$) then

Repeat (k_1에 할당된 모든 블록 j에 대해서)

if ($j \notin N_i^1$ and $j \notin N_i^2$) then

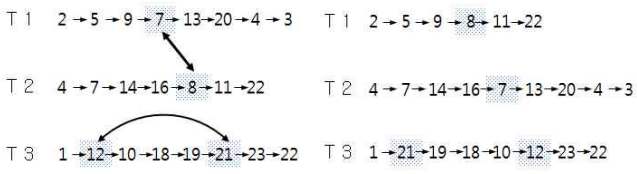
블록 j와 블록 i를 포함하여 두 블록 사이의 블록들의 순서를 역순으로 재배치, 블록 운반 계획에 문제가 발생하지 않으면 해를 이웃해에 추가

else if ($w' \leq c_{k_1}$ and $\sum_{j \in N} x_{ij}^k = 0$) then

Repeat (k_1에 할당된 모든 블록 j에 대해서)

if ($j \notin N_i^1$ and $j \notin N_i^2$) then

블록 j와 블록 i 다음 순서의 블록들의 트랜스포터를 교체, feasible한 해를 이웃해에 추가



(a) Initial (b) Neighborhood

Fig 5 2-Opt operator

경로 생성 알고리즘에서 블록당 한 대의 차량을 배치시켜 초기해를 구성하고 3.2의 3가지 operator로 이웃해를 탐색하는 것을 제안한다. 또한 타부 서치를 여러 번 반복하게 하여 탐색 영역을 넓혀 해 개선 가능성을 높이고자 한다. 3.2의 타부 서치를 단순히 반복하는 알고리즘(H1), 일정한 확률로 개선이 발생하지 않는 경우에도 해로 대치될 수 있도록 하여 반복하는 알고리즘(H2), 초기해를 재구성하여 반복하는 알고리즘(H3)의 세 가지 알고리즘을 제안한다. 경로 생성 알고리즘의 Step 1에서는 차량을 크기에 따라 정렬하였으나 H3 알고리즘은 일정 기간 이웃해 탐색에 개선이 없으면 랜덤하게 차량을 배치하여 생성된 해를 초기해로 재구성하는 방법이다.

4. 실험 및 결과 분석

본 논문에서 제시한 수리적 모형은 MIP 모형으로 컴퓨터를 이용하지 않으면 최적해를 찾기가 어려우므로 ILOG CPLEX 12.0를 이용하여 초기해를 탐색한다. 그리고 제안된 휴리스틱 알고리즘 및 타부 서치 알고리즘은 C#으로 구현하였다. 실험에 쓰인 트랜스포터와 블록 데이터는 Jeong(2006)에서 제시한 현장 데이터를 기준으로 작성하였다.

최적해와 제안한 알고리즘의 수행 결과 및 계산 시간을 Table 1과 같이 나타낼 수 있다. 최적해와 알고리즘 결과 값을 비교해보면 이동 블록 수가 10일 경우 평균 1의 차이가 있으며 5개 중 2개의 경우에서는 알고리즘으로 최적해를 구할 수 있었다. 이동 블록 수가 20일 경우 평균 9의 차이를 보였으며 최적해에 근사한 해를 출력하는 것으로 볼 수 있다. 또한, 10개에서 20개로 이동 블록 수가 증가할 때 최적해 탐색에 소요되는 시간이 크게 증가한데 반해 휴리스틱 알고리즘의 수행 시간은 이동 블록 수가 10, 20개인 실험에서 모두 1초를 넘지 않은 것으로 나타나 본 알고리즘의 탐색 시간의 효율성을

Table 1 Experimental Results

Case	Optimal Solution		H1		H2		H3	
	Sol	Time (sec)	Sol	Time (sec)	Sol	Time (sec)	Sol	Time (sec)
10_1	71	7.0	74	0.80	82	0.33	72	0.2
10_2	76	9.4	78	0.88	78	0.36	78	0.2
10_3	70	9.1	75	0.83	70	0.34	70	0.2
10_4	84	10.4	85	0.87	85	0.34	84	0.2
10_5	76	9.4	78	0.82	82	0.35	78	0.2
20_1	104	87.9	113	1.75	113	0.90	113	0.5
20_2	88	146.6	103	1.23	101	0.71	98	0.6
20_3	101	197.8	109	1.45	109	0.69	107	0.6
20_4	101	451.8	113	1.46	115	0.79	111	0.6
20_5	101	621.3	113	1.34	115	0.73	110	0.6
50_1	NA		220	27.5	171	53.8	187	4.2
50_2	NA		304	33.0	278	49.6	308	8.9
50_3	NA		274	41.7	244	45.3	253	7.1
50_4	NA		265	39.3	263	48.3	266	9.0
50_5	NA		331	33.5	311	48.2	328	6.9

Table 2 Experimental Results

Case	Optimal Solution		H1		H2		H3	
	Sol	Time (sec)	Sol	Time (sec)	Sol	Time (sec)	Sol	Time (sec)
200_1	NA		511	141.4	464	446.9	471	63.3
200_2	NA		646	83.81	514	299.1	495	48.8
200_3	NA		547	134.7	468	408.7	489	47.9
200_4	NA		575	121.1	514	312.5	491	45.3
200_5	NA		565	119.7	520	300.7	491	49.5
300_1	NA		787	143.8	789	680.0	787	74.1
300_2	NA		614	171.1	605	696.6	613	76.9
300_3	NA		595	169.1	618	576.8	598	69.0
300_4	NA		684	163.8	646	647.8	644	89.7
300_5	NA		732	176.2	693	616.9	734	89.3
500_1	NA		901	293.2	904	1571.0	904	129.7
500_2	NA		853	328.5	864	1559.3	859	115.7
500_3	NA		904	325.8	910	1698.4	906	154.5
500_4	NA		957	306.5	968	1645.4	963	142.8
500_5	NA		901	361.0	910	1591.4	905	140.9

보여준다. 이동 블록 수가 30를 넘어갈 때는 최적해를 구하는 것이 불가능하여 구해진 해의 품질을 비교할 수 없었다. 그러나 이동 블록 수가 10, 20일 때 실험 결과로 보아 그 결과가 나쁘지 않을 것으로 예상된다. 최적해를 구할 수 없었던 실험 결과까지 포함하여 제안한 3개의 휴리스틱 알고리즘들을 비교해보면 블록 수가 200개 이하일 때, H2 > H3 > H1, 200~300개일 때는 H3 > H2 > H1 순으로 좋은 해를 탐색하였으며 500개일 때는 H1 > H3 > H2 순으로 좋은 해를 탐색하였다. 여기서 300개까지 H1의 결과 값은 H2나 H3에 비해 나빴으며 계산 시간도 H3에 비해 오래 걸리는 것을 확인할 수 있었으며 500개부터는 가장 좋은 해를 탐색했다. 일반적으로 변형 기법에 비해 좋은 해를 낼 수 없었지만 블록의 수가 많아질수록 단순히 타부 서치를 반복하는 것이 가장 좋은 해를 탐색하는 방법이다. H2는 대체로 가장 좋은 해를 탐색하였지만 시간이 다른 알고리즘들에 비해 오래 걸린다는 것을 볼 수 있다. 반면 H3는 모든 경우에서 가장 짧은 탐색 시간을 가졌고 H1, H2의 해와 비슷한 답을 제시하거나 특정 블록 개수군의 데이터에서는 가장 우수한 답을 내기도 하였다. 이는 H2에서 한 번 해 탐색 방향이 잘못되면 계속해서 나쁜 이웃해를 탐색할 가능성이 있고 H3는 초기해만 바뀔 뿐 해 탐색이 좋은 방향으로만 이루어지기 때문인 것으로 보인다. 그러므로 시간 대비 효율이 가장 좋은 알고리즘은 H3이고 실제 현장에서는 수백 개의 블록이 이동하므로 현장에 적용하기 가장 적합한 알고리즘일 것으로 생각된다. 그러나 가장 오랜 시간이 걸린 알고리즘 H2의 경우도 200개의 블록 이동을 탐색할 때 걸린 시간이 6분을 넘지 않았으므로 위의 알고리즘들이 빠른 시간 내에 우수한 해를 탐색할 수 있는 것으로 여겨진다.

5. 결 론

기술의 발달로 조선소에서 결합할 수 있는 트랜스포터를 사용할 수 있게 되었지만 아직까지 트랜스포터를 연결을 고려하여 효율적으로 운영하는 연구는 거의 이루어지지 않았다.

본 논문에서는 트랜스포터의 결합에 따라 용량 제약 및 시간 제약이 변하는 것을 고려하였으며 조선소에서 발생하는 블록 이동을 적시에 수행하기 위해 트랜스포터를 스케줄링하는 문제를 연구하였다. 조선소의 블록 운반 상황에 맞춰 문제를 설정하고 그에 따른 수리적 모형과 휴리스틱 해법을 제시하였다. 그리고 수리적 모형 및 제안한 해법을 이동해야 할 블록 수에 따라 다양한 데이터로 실험한 후 분석을 통해 해법의 효율성을 검증하였다. 실험 분석 결과, 제안한 휴리스틱으로 짧은 시간 내에 우수한 해 탐색이 가능하므로 동적 환경의 트랜스포터 일정 계획에서 이용될 수 있을 것으로 기대된다.

향후 블록 적치장의 블록 반출입 계획과 공정 계획에 트랜스포터 일정 계획을 연계하여 계획을 하게 되면 전체 블록 흐름상 더 좋은 해를 낼 것으로 보이며 그에 따른 추가적인 연구가 필요할 것으로 보인다.

References

[1] Chung, C. Y. and Shin, J. Y.(2012), “Efficient Yard Tractor Control Method for the Dual Cycling in Container Terminal”, Journal of navigation and port research, Vol. 36, No. 1, pp. 69-74.

[2] Dror, M., Trudeau, P.(1990), "Split delivery routing", Naval Research Logicsticcs, Vol.37, No.3, pp. 383-402.

[3] Desaulniers, G.(2010), "Branch-and-price-and-cut for the split delivery vehicle routing problem with time windows". Operations Research, Vol.58, No.1, pp. 179-192.

[4] Dror, M., Laporte, G., Trudeau, P.(1994), "Vehicle routing with split deliveries", Discrete Applied Mathematics, Vol.50, No.3, pp. 239-254.

[5] Ho, S., Haugland, D.(2004). "A tabu search heuristic for the vehicle routing problem with time windows and split deliveries". Computers and Operations Research, Vol.31, pp. 1947-1964.

[6] Jeong, Y. S.(2006). Block Transportation Management System, Thesis(master), pp.8-9.

[7] Kim, J. H. et al(2008), “A Development of Floating Dock Control Simulator for Skid Launching System“, Journal of navigation and port research, Vol. 32, No. 1, pp. 1-7.

[8] Lee, W. S. et al(2008), “Transporter Scheduling for Dynamic Block Transportation Environment”, IE Interfaces, Vol. 21, No. 3, pp. 274-282.

[9] Lee, W. S. et al(2009), “Transporter Scheduling Based on a Network Flow Model for Dynamic Block Transportation Environment”, IE Interfaces, Vol. 22, No. 1, pp. 63-72.

[10] Park, C. K. and Seo, J. Y.(2012), ″A GRASP approach to transporter scheduling and routing at a shipyard ″, Computers & Industrial Engineering, Vol. 63, No. 2, pp. 390-399.

[11] Shin, J. Y. and Kwon S. C.(2009), “Effective Operation Strategies for Pooling Yard Tractors m Container Terminals”, Journal of navigation and port research, Vol. 33, No. 6, pp. 401-407.

[12] Yim, S. B. et al(2008), “Optimal Block Transportation Scheduling Considering the Minimization of the Travel Distance without Overload of a Transporter”, Journal of the Society of Naval Architects of Korea, Vol. 45, No. 6, pp. 646-655.

원고접수일 : 2013년 12월 26일
심사완료일 : 2014년 5월 20일
원고채택일 : 2014년 5월 22일