韓國 컴퓨터情報學會 論文誌
第9卷 第4號, 2004. 12.

2004-9-4-2-4

반복적 개선 탐색을 이용한 최적 선석 및 크레인 일정계획 *

황 준 하**

An Iterative Improvement Search for the Optimal Berth and Crane Scheduling

Junha Hwang **

요 약

컨테이너 터미널에서의 선석 및 크레인 일정계획은 일정 기간 동안 입항 예정인 선박들을 대상으로 선석을 배정하고 접안 시기와 기간을 결정하며, 또한 각 선박별로 컨테이너를 싣고 내릴 크레인을 배정하되 각 크레인의 서비스 시작과 완료시간을 지정하는 전 과정을 포함한다. 이 문제는 기본적으로 다양한 제약조건을 만족해야 하는 제약만족 탐색 문제인 동시에, 각 선박의 희망 입출항 시간을 최대한 존중하면서 터미널의 운영 비용을 최소화할 수 있는 계획을 도출해야 하는 최적화 문제이기도 하다. 본 논문에서는 이 문제를 효과적으로 해결하기 위해 반복적 개선 탐색의 틀 내에서 제약만족 탐색기법을 적용하는 방안을 제시하고 있다. 실제 컨테이너 터미널에 대한 실험 결과 기존 알고리즘을 적용했을 때보다 더 좋은 계획을 수립할 수 있음을 확인하였다.

Abstract

The berth and crane scheduling problem in a container terminal encompasses the whole process of assigning berth to each ship, determining the duration of berthing, assigning container cranes to each ship, and determining the specific start and end time of each crane service, for all the ships scheduled to be arriving at the terminal during a certain scheduling horizon. This problem is basically a constraint satisfaction problem in which all the constraints should be satisfied. However, it is also an optimization problem because the requested arrival and departure time should be met for as many of the scheduled ships as possible, while the operation cost of the terminal should be minimized. In this paper, I present an effective approach to solving this problem, which combines both constraint satisfaction search and iterative improvement search. I test this method on a real world container terminal problem and the results show that the method can produce better results than any other existing method.

▸ Keyword : 반복적 개선 탐색, 제약만족 탐색기법, 선석 및 크레인 일정계획

• 제1저자 : 황준하
• 접수일 : 2004.10.11, 심사완료일 : 2004.11.19
* 본 논문은 금오공과대학교 학술연구비에 의하여 연구된 논문임.
** 금오공과대학교 컴퓨터공학부 조교수

I. 서 론

오늘 날 국제적 교역이 활발해지고 교역량이 점점 더 확대됨에 따라 수출입 물량의 원활한 처리 및 서비스 향상의 중요성이 날로 증대되고 있다. 선박을 이용한 화물 운송은 운송 수단들 중 가장 큰 비중을 차지하고 있는 운송 수단으로서 항만 터미널의 업무를 자동화하는 것이 효율적인 물류 처리를 위한 중요한 과제라 할 수 있다[1, 2]. 특히 선석 및 크레인 일정계획의 수립 결과는 각 선사의 비용을 줄이는 데 큰 영향을 미칠 뿐 아니라 항만 터미널의 비용을 최소화하기 위한 주요 요소로 인식되고 있어, 보다 최적의 계획을 빠른 시간 내에 수립할 수 있는 방안이 요구되고 있다.

항만 터미널에서의 선석 계획은 일정 기간 동안 입항 예정인 선박들을 대상으로 접안 위치를 배정하고 접안 시기와 기간을 결정하는 것을 말하며, 크레인 일정계획은 각 선박별로 컨테이너를 운반할 크레인을 배정하고 각 크레인의 서비스 시작 시간과 완료 시간을 지정하는 것을 말한다. 그런데 선석 계획 결과는 크레인 일정계획에 영향을 미치게 되고 역으로 크레인 일정계획의 수립이 어려울 경우 선석 계획에 영향을 미치게 되므로 선석 계획과 크레인 일정계획은 동시에 수립되는 것이 바람직하다.

선석 및 크레인 일정계획 수립 시 선박의 접안 위치는 싣고 내릴 대상 컨테이너들의 장치장과 가능한 가까운 곳으로 하는 것이 좋고, 접안 기간은 작업량에 합당한 적정 수의 컨테이너 크레인(Container Crane: CC)을 각 선박에 배정해 줌으로써 선사의 희망 입출항 시간을 지킬 수 있도록 하는 것이 좋다. 그러나, 주어진 계획기간 내에 여러 선박들이 있을 경우 선박들 상호간의 시간적 또는 공간적 간섭 현상 때문에 모든 선박의 요구조건을 다 만족시키는 계획을 도출해 내는 것이 어려운 경우가 많다. 한 선박의 접안 위치 및 접안 기간의 변경은 시공간적으로 인접한 이웃 선박들에 간섭을 일으켜 그들의 시공간적 위치 변경을 초래하게 되고 이런 현상이 결국 전 선박으로 파급되기도 한다. 또한, 각 CC는 그 이동 범위가 제한되어 있기 때문에 선박의 접안 위치가 바뀔 경우 배정 가능한 CC가 달라지게 되며 그에 따라 소요 작업 시간이 달라지게 되고 선박의 접안 기간도 영향을 받게 된다. 선석 및 크레인 일정계획은 이와 같이 여러 가지 제약조건들이 서로 복잡하게 영향을 주고받는 고난도의 문제이다.

이러한 종류의 문제는 소위 제약조건 만족 문제(constraint satisfaction problem: CSP)의 형태로 정형화될 수 있고, 그럴 경우 제약조건을 반영하여 탐색 공간을 줄여 나감으로써 해를 효율적으로 찾아 주는 제약만족 탐색(constraint satisfaction search)기법을 활용하여 해결할 수 있다. 그러나 본 논문의 선석 및 크레인 일정계획 문제에는 다양한 제약조건 외에도, 가능하면 지정된 선호 위치에 가깝게 접안 위치를 선정해야 한다든지 선사의 희망 입출항 시간을 최대한 준수해야 하는 등 최적화적 요소가 포함되어 있다. 그런데 제약만족 탐색기법은 제약조건을 만족하는 해를 효율적으로 찾아 줄 뿐, 직접적으로 최적해를 찾아주지는 못한다는 문제가 있다. 그렇다고 이 문제를 보통의 최적해 탐색 기법만으로 해결하고자 할 경우에는 제약조건을 위배하지 않는 해를 찾는 것 자체가 어려워지게 되어 탐색이 효율적이지 못하게 된다. 흔히, 이러한 종류의 문제를 제약조건 만족 및 최적화 문제(constraint satisfaction optimization problem: CSOP)라 부르며[3], 일반적인 CSP나 최적화 문제와는 다른 방식의 해결 방안을 필요로 한다.

기존 연구에서는 현재해를 개선하기 위한 한 가지 방법으로 제약만족 탐색기법을 위한 휴리스틱 교정 기법을 제시한 바 있다[4]. 본 논문에서는 보다 최적의 해를 도출하기 위한 방안으로 기존 연구를 보완하여 반복적 개선 탐색의 틀 내에서 제약만족 탐색기법을 적용하는 방법을 제시하고 있다. 실제 컨테이너 터미널의 다양한 선석 계획 자료를 대상으로 실험한 결과 본 논문이 제시한 방법이 기존의 방법보다 더 좋은 계획을 신속하게 수립할 수 있음을 확인하였다.

II. 대상 문제

2.1 선석 및 크레인 일정계획

선석 계획은 월간 선석 계획과 주간 및 일일 선석 계획으로 나뉘어진다. 월간 선석 계획은 선사와 항만 터미널의 협의 계약인 장기 Calling Schedule과 선사가 보내온 월간 입항계획서를 바탕으로 작성된다. 월간 선석 계획에서는 미

래의 불확실성을 수용한 상태에서 일단 선석을 가 배정해 두되 보다 정확한 입항정보가 들어올 때마다 기존 계획을 수정함으로써 계획을 단계적으로 확정해 간다. 주간 및 일일 선석 계획은 매일매일 접수되는 정보를 수시로 반영하여 실제로 선석을 배정하는 계획이다. 주간 선석 계획은 월간 선석 계획에 제시된 선석 운영계획 및 각 선박별 예상 처리 화물량에 근거하여 CC를 할당하고 선석을 배정한다. 일일 선석 계획에서는 보다 정확한 화물 처리작업 소요시간의 추정치에 근거하여 필요 시 CC를 다시 할당하고 선석 배정을 확정한다. 선석 배정시 각 선박은 터미널 운영 효율상 유리하도록 취급 컨테이너들이 장치되어 있는 곳과 가까운 위치에 접안시키는 것이 좋다. 물론, 선석 배정은 계획 기간 내의 모든 선박들에 대해 서로 접안 위치와 접안 기간이 겹치는 부분이 없어야 한다.

크레인 일정계획은 각 선박에 대해 선석 계획 결과로 부여받은 접안 위치에서 입항 시간부터 시작하여 출항 시간 전까지 모든 서비스를 마칠 수 있도록 CC를 할당하는 계획을 말한다. CC들은 전체 선석에 걸쳐 안벽과 평행하게 설치된 궤도상에서 움직이면서 선박에 대한 서비스를 수행한다. 각 CC는 한정된 범위 내에서 이동이 가능하므로 선박의 접안 위치에 따라 배정 가능한 CC가 달라진다. 한 선박에 대해서는 대개 2~5기의 CC를 배정하여 각 CC가 선박의 특정 구역을 담당하게 한다. CC의 배정 역시 서로 다른 CC들이 시간 공간적으로 서로 겹치는 일이 없도록 되어야 한다. 그런데 CC의 배정에 따라 선박의 접안기간이 달라질 수 있고 그럴 경우 다른 선박의 선석 배정에도 영향을 주기 때문에, 선석 계획과 크레인 계획은 별도가 아니라 동시에 이루어지는 것이 바람직하다.

본 논문은 부산 신선대 컨테이너 터미널(PECT: Pusan East Container Terminal)의 주간 선석 및 크레인 일정계획 문제를 대상으로 하고 있다. 그림 1은 주간 선석 및 크레인 일정계획의 예로서 가로축은 안벽에 해당하고 세로축은 시간을 의미한다. 각 사각형은 해당 선박이 가로축 길이 만큼의 안벽을 세로축 만큼의 기간 동안 점유함을 표시하고 있다. 따라서 그림 1과 같이 어떠한 사각형도 서로 겹치지 않아야 시행 가능한 계획이 된다. 뿐만 아니라 월간 계획 시 결정된 선호 위치를 비롯하여 희망 입항 시간과 희망 출항 시간을 최대한 준수하는 것이 좋다.

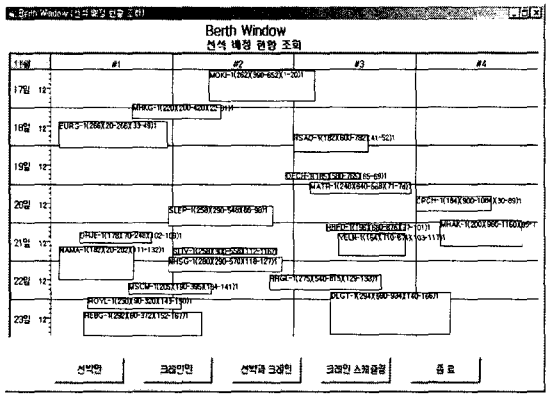

그림 1. 선석 계획의 예
Fig. 1 An example of berth scheduling

2.2 관련 연구

선석과 크레인 일정계획을 동시에 수립하는 방안에 관해서는 국내외적으로 현재까지 연구된 사례가 드문 형편이며, 대부분 선석 일정계획과 크레인 일정계획을 각기 독립적인 문제로 다루고 있다. Daganzo는 각 접안 대상 선박별로 각 시간대에 할당할 CC의 수를 결정하는 휴리스틱 기법을 제시하였다[5]. 그리고 Peterkofsky 등은 여러 척의 선박이 대기하고 있는 상태에서 여러 기의 크레인을 이용하여 총 접안 시간의 가중합을 최소화하기 위한 선박의 작업 순서 및 크레인 할당 수를 결정하는 문제를 다루었다[6].

윤철영 등은 컨테이너 터미널의 선석 및 크레인 운영을 최적화하는 모델을 소개하였다[7]. 이 연구에서는 선석당 크레인수를 2기에서 5기로 증가시킴으로써 선석의 효율을 높일 수 있음을 보였다. Park과 Kim은 선석 및 크레인 일정계획의 최적화를 위한 수리 모형 및 Lagrangian relaxation 방법을 제시한 바 있다[8]. 그러나 이상의 방법들은 대부분 수학적 모델에 기반을 둔 접근 방법들로서 문제의 규모가 커짐에 따라 계산량이 기하급수적으로 늘어나게 되므로 현실적 규모의 문제를 풀기가 어렵다는 단점이 있으며, 본 논문의 대상 문제와 같이 수리적 모형으로 표현하기 어려운 복잡한 제약조건 또는 목적함수를 포함할 경우 적용 자체가 불가능하다.

류광렬 등은 제약만족 탐색기법 적용 시 반복적인 개선을 위해 휴리스틱 교정기법을 사용하여 변수값의 할당 순서를 재지정함으로써 효과적인 탐색이 가능하도록 하였다[4]. 본 논문에서는 이 연구를 바탕으로 반복적 개선 탐색이 보다 효율적으로 진행될 수 있도록 하기 위한 방안을 제시하고 있다.

III. 제약만족 탐색 기법의 적용

3.1 제약만족 탐색 기법의 개요

제약만족 탐색 문제는 변수들의 집합과 도메인(domain)들의 집합 및 변수들간의 제약조건으로 구성되는데[9], 도메인이란 각 변수가 취할 수 있는 값들의 집합을 말한다. 제약만족 탐색 문제의 해를 구하는 과정은 모든 변수들에 대해 그들 간에 주어진 제약조건을 모두 만족하도록 각 도메인으로부터 적절한 변수값을 지정하기 위한 탐색 과정이 된다. 이 과정은 주로 깊이우선 탐색(depth-first search) 방식으로 이루어진다. 즉, 소정의 순서에 따라 각 변수에 해당 도메인으로부터 값을 찾아 할당해 나가되, 매 할당 시 제약조건 만족 여부를 검사하여 제약조건을 만족하면 다음 변수로 넘어가서 변수값 할당을 계속하고 그렇지 못하면 그 변수의 도메인에서 다음 값을 할당한다. 만약 도메인 내의 모든 값들이 전부 제약조건을 위배한다면, 직전의 변수로 되돌아가(backtracking) 그 변수의 값을 다음 값으로 바꾼다. 직전 변수에서도 제약조건을 만족하는 값을 찾을 수 없다면 백트래킹을 계속해야 하고, 그렇지 않다면 값을 할당한 후 순서상의 다음 변수로 넘어가서 동일한 과정을 계속 수행한다. 이런 방식으로 마지막 변수까지 값 할당에 성공하면 하나의 해를 찾은 것이 된다.

그러나 이렇게 탐색 공간 전체를 검색하는 깊이우선 탐색 방식만으로는 탐색공간의 규모가 클 경우 해를 찾기 어렵다. 따라서 보통 제약만족 탐색기법에서는 제약조건을 이용하여 탐색공간을 줄이는 방안으로서, 전향검사(forward checking)를 통한 도메인 축소(domain reduction) 기법을 활용하고 있다. 이 기법에 따르면 특정 변수에 값이 할당될 때마다 아직 값이 할당되지 않은 나머지 모든 변수 각각에 대해 이미 값이 할당된 변수들이 그 값들을 유지할 경우 이들과 제약조건 상 모순되는 값들을 파악하여 해당 변수의 도메인으로부터 미리 제거함으로써 예측 가능한 백트래킹을 사전에 방지하게 된다.

제약만족 탐색기법의 효율을 좌우하는 또 다른 요소로는 변수의 탐색 순서 및 각 도메인으로부터 선택하는 변수값의 탐색 순서를 들 수 있다. 대상 문제를 잘 분석하여 이들 순서를 적절히 지정하면 백트래킹이 줄어들어서 그렇지 못한 경우보다 훨씬 빨리 해를 찾을 수 있게 된다. 변수 순서 및 변수값 순서 지정 휴리스틱은 제약만족 탐색 문제뿐만 아니라 본 논문의 대상 문제와 같은 제약만족 및 최적화 문제 해결을 위해서도 효과적으로 적용될 수 있다.

3.2 문제의 표현

본 논문의 대상 문제에서는 각 선박의 입항 시간, 출항 시간, 접안 위치, 각 CC의 각 선박에 대한 서비스 여부, 각 CC의 서비스 시작 시간, 각 CC의 서비스 완료 시간이 변수로 표현된다. 선박의 입출항 예정 시간과 CC의 서비스 시작 및 완료 시간에 대한 변수의 도메인은 전체 일정계획 대상기간 내에 포함된 1시간 단위 간격의 값들을 포함한다. 선박의 접안 위치 즉 선수의 위치를 나타내는 변수는 안벽의 총 길이에 대한 10m 간격의 1차원 좌표값들을 도메인으로 가진다. 마지막으로, 각 선박에 대한 각 CC의 서비스 여부는 0과 1로 표시된다.

고려해야 할 주요 제약조건들로는 다음과 같은 것들이 존재한다. 접안 위치와 관련하여 CC의 이동 가능 범위 때문에 안벽의 시작과 끝 부분에 너무 가깝게 선박이 접안할 수 없다는 제약이 있으며 선박들 상호간에는 시공간적으로 적정한 안전 간격을 지키도록 되어 있다. 각 선박의 입항 시간과 출항 시간은 CC의 서비스 시작 및 완료 시간과 제약 관계에 있다. 즉, 한 선박에 배정된 CC들은 모두 그 서비스 시작 시간이 그 선박의 접안 이후라야 하고, 완료 시간은 출항 이전이어야 한다. 이 외에도 총 20여 가지의 제약조건들이 존재한다.

대상 문제의 목적함수는 식 (1)과 같이 표현될 수 있다. 여기서 $P(i)$, $EP(i)$는 각각 선박 i의 접안 위치와 선호 접안 위치를 의미하고 $A(i)$, $D(i)$는 각각 선박 i의 입항 시간과 출항 시간을 의미하며 $ETA(i)$와 $ETD(i)$는 각각 해당 선박의 희망 입항 시간(Estimated Time of Arrival)과 희망 출항 시간(Estimated Time of Departure)을 의미한다. 즉, 선박의 접안 위치는 컨테이너들이 장치되어 있는 선호 위치와 가까운 위치에 접안시키는 것이 좋고 입항 시간은 희망 입항 시간을 최대한 준수하는 것이 좋으며 출항 시간은 희망 출항 시간보다 늦어지지 않는 것이 좋다.

$$Obj = Minimize\Big(\sum_{i=1}^{n}(|EP(i)-P(i)| + |ETA(i)-A(i)| + Max(0, D(i)-ETD(i)))\Big) \cdots\cdots (1)$$

3.3 변수 및 변수값 순서 지정 휴리스틱

제약조건 만족 문제를 해결하는 데 변수 및 변수값의 탐색 순서가 큰 영향을 미치는 것은 분명한 사실이지만 특정 문제에 있어서 변수 및 변수값의 탐색 순서를 어떻게 하는 것이 탐색의 양을 줄이는데 가장 효과적인지 판단하기란 쉽지 않다. 대상 문제마다 특성이 달라 일반적인 휴리스틱보다는 그 문제에 적합한 변수 및 변수값 순서 지정 휴리스틱을 개발하여 사용하는 것이 보통이다[10].

본 논문에서는 각 변수들의 탐색 순서를 실험적으로 결정하였으며 기본적인 탐색 순서는 접안 위치, 입항 시간, 출항 시간, 크레인 시작 시간 그리고 크레인 완료 시간 변수 순으로 진행된다. 그리고 각 변수들 내에서 선박별 순서는 희망 입항 시간 순으로 지정하는 휴리스틱을 사용하였다. 예를 들면 희망 입항 시간이 가장 빠른 선박의 접안 위치가 가장 먼저 결정되고 희망 입항 시간이 다음으로 빠른 선박의 접안 위치가 결정되는 방식으로 모든 선박의 접안 위치가 결정된다. 그리고 다음으로 각 선박의 입항 시간이 희망 입항 시간 순으로 결정된다.

변수값 순서 지정 휴리스틱은 대상 문제의 최적화적 요소를 고려하여 식 (1)의 목적함수 값을 최소화할 수 있는 값을 우선적으로 지정하는 방식을 사용하였다. 접안 위치의 경우 선호 위치로부터 가까운 위치를 먼저 할당하게 되는데, 예를 들어 어떤 선박의 선호 위치가 500인 경우 (500, 501, 499, 502, 498, ...)의 순으로 값 할당을 시도하게 된다. 이와 마찬가지로 입항 시간 역시 희망 입항 시간과 가까운 값들을 먼저 지정한다. 단, 출항 시간은 희망 출항 시간보다 작은 값들 중에서 희망 출항 시간과 가까운 값들을 먼저 지정하게 되는데 이는 출항 시간의 경우 희망 출항 시간보다 늦어지는 경우에만 목적함수가 나빠지게 되기 때문이다. 이와 같은 휴리스틱은 목적함수의 최적화 측면뿐만 아니라 제약조건을 모두 만족하는 하나의 해를 찾아내는 데도 상당히 효과적인 것으로 판단된다. 왜냐하면 대상 문제의 특성 상 비록 크레인 일정계획 등 세부 사항들을 고려하지는 못했다 하더라도 상위 계획 단계에서 경험이 반영된 선박별 선호 위치 등이 가 배정되어 있는 상태이므로, 선호 위치 또는 희망 입항 시간과 가까운 곳부터 차례로 변수값을 지정하는 것이 백트래킹을 줄일 수 있는 효과가 있을 것으로 판단되기 때문이다.

3.4 변수값 순서의 재정렬

기존 연구 [4]에서는 제약만족 탐색의 성능을 향상시키기 위해 휴리스틱 교정 방법을 사용하여 변수값의 탐색 순서를 재정렬한 후 또 다시 제약만족 탐색을 수행하는 방법을 사용하였으며, 본 논문 역시 Ⅳ장에서 설명할 반복적 개선 탐색 알고리즘을 위해 이 방법을 사용하고 있다. 따라서 본 절에서는 [4]에서 사용했던 방법에 대해 간략히 소개하며 이 방법의 문제점에 대해 설명한다.

제약만족 탐색을 수행할 경우 선호 위치, 희망 입항 시간, 희망 출항 시간으로부터 가까운 값부터 할당하게 되므로 첫 번째로 도출된 해는 최적해와는 거리가 멀다 하더라도 어느 정도 좋은 해가 도출될 것으로 기대할 수 있다. 그러나 값을 할당하는 선박의 순서가 미리 고정되어 있어 비교적 순서가 뒤인 선박들은 자신의 선호 위치 또는 희망 입항 시간에 배정되지 못하는 현상이 발생한다.

이와 같은 상황을 피하기 위해서는 우선순위가 높은 선박의 접안 위치나 입출항 시간을 지정할 때 우선순위가 낮은 선박의 선호 위치나 희망 입출항 시간까지 고려하는 것이 바람직하다. 따라서 현재해로부터 선박들 간의 계획 결과를 분석하여 우선순위가 낮은 선박들이 선호 위치나 희망 입출항 시간에 배정될 수 있도록 우선순위가 높은 선박들의 변수값 순서를 재정렬한 후 또 다시 제약만족 탐색을 수행하는 과정을 반복하고 있다. 만약 현재해의 분석 결과 더 이상 변수값 순서 조정에 의한 해의 개선이 불가능하다고 판단될 경우 알고리즘은 종료하게 된다.

이와 같이 현재해를 분석하여 변수값의 탐색 순서를 변경한 후 제약만족 탐색을 재수행함으로써 더 좋은 해를 찾을 수 있을 것으로 기대된다. 그러나 현재해를 분석할 때 모든 선박들에 대해 변경 가능한 모든 조합을 대상으로 개선 여부를 조사하는 것 자체가 원문제의 난이도와 유사한 탐색을 요구하게 되므로, 불가피하게 개별 선박 단위로 이동 가능성 여부만을 조사하였다. 따라서 실제로는 현재해보다 더 좋은 해가 존재함에도 불구하고 해의 개선이 불가능하다고 판단함으로써 알고리즘이 조기에 종료되는 상황이 발생할 수 있다. 실험에 의하면 대부분 제약만족 탐색을 2, 3회 정도 재수행한 후 더 이상 해의 개선이 어렵다고 판단하여 프로그램이 종료됨을 확인할 수 있었다.

Ⅳ. 반복적 개선 탐색 알고리즘

4.1 목적함수의 제약조건화

하나의 제약만족 탐색을 수행하는 도중에 새로운 해가 도출되었다고 가정하자. 그렇다면 최적화 문제의 경우 그 다음에 도출해야 할 해는 최소한 방금 찾은 해보다는 더 좋은 해이어야만 한다. 일반적으로 깊이우선 탐색 도중에 특정 노드에서의 최소 비용이 현재까지 찾은 가장 좋은 해보다 클 경우에는 그 노드로부터 시작하는 탐색은 고려하지 않도록 함으로써 불필요한 탐색 공간을 줄일 수 있는데, 이와 같은 탐색 기법을 Branch & Bound라고 부른다[11].

그림 2는 변수 3개(x_1, x_2, x_3)로 이루어져 있고 각각 0과 1의 값을 도메인으로 가지고 있는 제약만족 탐색 및 최적화 문제에 대해 깊이우선 탐색 방식으로 수행되는 Branch & Bound 탐색의 예를 보인 것이다.

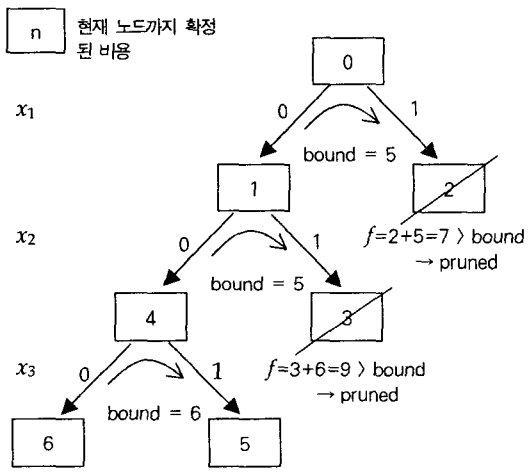

그림 2. Branch & Bound 탐색의 예
Fig. 2 An example of Branch & Bound

각 노드의 값은 그 때까지 결정된 값들에 의해 계산된 비용을 의미하고 각 분기(branch) 위의 값은 해당 변수에 할당된 값을 의미한다. 각 노드에서의 최소 비용 f는 $(g+h)$로 구할 수 있는데 g는 그 때까지 결정된 비용을 의미하고 h는 그 노드로부터 마지막 노드가 결정될 때까지 추가될 것으로 예상되는 최소 비용을 의미한다. 그렇다면 그림 2에서와 같이 특정 노드의 예상 최소 비용 f가 현재까지 찾은 가장 좋은 해의 비용(bound)보다 같거나 큰 경우, 더 이상 그 노드 이하의 탐색은 불필요한 탐색으로 간주하고 이하의 탐색을 생략할 수 있다. 결과적으로 불필요한 탐색 공간을 줄임으로써 보다 빠르게 더 좋은 해의 도출이 가능하게 된다.

본 논문의 대상 문제에도 이와 같은 탐색 방법을 그대로 적용할 수 있다. 적용 방법은 식 (2)와 같이 "새로운 해의 목적함수 값이 현재해보다 작다"라는 제약조건을 추가하는 것이다. 여기서 *Obj*는 새롭게 찾아야 하는 해의 목적함수 값을 의미하며 *CurObj*는 현재해의 목적함수 값을 의미한다.

AddConstraint (*Obj* 〈 *CurObj*) ····· (2)

내부적으로는 그림 2에서 설명한 바와 같이 특정 노드에서의 예상 최소 목적함수 값을 구한 후 그 값이 현재까지의 가장 좋은 해의 목적함수 값보다 크다면 이하의 탐색은 고려 대상에서 제외하면 된다. 이 때 현재 노드로부터 추가될 최소 비용 h는 아직 결정되지 않은 각 변수의 남아 있는 도메인들을 대상으로 쉽게 계산될 수 있다. 목적함수 계산에 동원되는 변수가 선수 위치, 입항 시간, 출항 시간이며 이 변수들이 변수 탐색 순서 상 가장 먼저 결정되게 되므로 불필요한 탐색 노드들을 탐색 초기에 효과적으로 제거할 수 있다.

4.2 반복적 개선 탐색 과정

본 논문에서 대상 문제의 해결을 위해 최종적으로 제안하는 방법은 변수값 탐색 순서를 재정렬하여 제약만족 탐색을 재수행하는 방법과 목적함수 값을 다시 제약조건으로 추가하는 방법을 결합하는 것으로서 수행 절차는 그림 3과 같다. 먼저 초기해를 생성하기 위해 제약만족 탐색을 수행하고 해를 분석하여 각 변수의 변수값 탐색 순서를 재조정하며 마지막으로 현재해의 목적함수 값을 사용하여 제약조건을 추가한 후 제약만족 탐색을 재수행한다. 즉, Ⅲ장 4절에서 설명한 변수값 탐색 순서의 재정렬 알고리즘을 적용함과 동시에 현재해의 목적함수 값을 사용하여 식 (2)와 같은 제약조건을 추가하는 것이다. 이렇게 함으로써 반복적 개선 탐색 과정에서 더 좋은 해가 발견되지 않을 것으로 판단되는 영역의 탐색을 생략함으로써 더 좋은 해가 보다 빨리 나올 수 있도록 유도하고 있다.

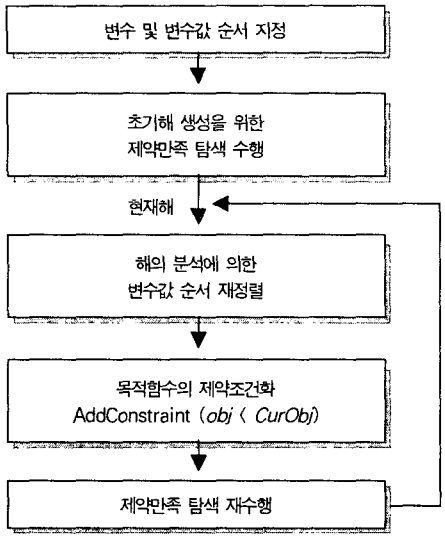

그림 3. 반복적 개선 탐색 과정
Fig. 3 Iterative improvement search process

Ⅳ장 1절의 알고리즘이 하나의 제약만족 탐색 내에서 새로운 해가 발견될 때마다 제약조건을 추가하는 반면에 이 방법은 반복적 개선 탐색 과정에서 새로운 제약만족 탐색을 재수행하기 전에 제약조건을 추가한다는 점에서 차이가 있다. 물론 단순히 목적함수 값만을 제약조건으로 추가한다면 이전 제약만족 탐색을 중단하는 의미가 없다. 따라서 이 방법에서는 목적함수 값을 제약조건으로 추가하는 것 외에 변수값 탐색 순서를 재정렬함으로써 현재해보다 더 좋은 해를 보다 빠른 시간 내에 도출할 수 있게 하였다.

Ⅴ. 실험 결과

실험 대상으로 사용한 자료는 부산 신선대 컨테이너 터미널의 주간 선석 계획 자료로서 총 25개의 데이터 Set을 사용하였으며 각 데이터마다 13~20개까지의 선박을 포함하고 있다. 제약만족 탐색기법의 구현을 위해 개발 도구인 ILOG Solver를 사용하였으며[12] 실험은 Pentium-IV 2Ghz PC에서 수행되었다.

첫 번째 실험에서는 처음 10개의 데이터 Set 각각에 대해 최초해를 구하기 위한 제약만족 탐색을 수행해 보았다. 특별한 변수의 탐색 순서 및 변수값 탐색 순서를 지정하지 않고, 변수는 임의의 변수를 선택하도록 하였으며 값은 가장 작은 값부터 할당하도록 하였다. 각 실험당 최대 수행 시간을 5분으로 지정하여 실험한 결과 어떤 데이터 Set에 대해서도 허용 시간 내에 모든 제약조건을 만족하는 해를 도출하지 못함을 확인하였다. 즉, 단순한 제약만족 탐색기법만으로는 최초해조차도 구하지 못함을 알 수 있다.

두 번째 실험은 변수의 탐색 순서 및 변수값 탐색 순서 지정 휴리스틱의 영향을 알아보기 위한 실험으로서 실험 결과는 〈표 1〉과 같다. 방법 1은 변수 선택 시 most-constrained variable 휴리스틱을 사용하였으며 변수값은 작은 값부터 할당하였다. 이 휴리스틱은 현재 도메인 집합의 크기가 가장 적은 변수를 우선적으로 선택하는 방법으로서 향후 탐색 시 고려해야 할 분기 계수(branching factor)를 줄일 수 있을 것으로 기대되는 방법이다. 방법 2에서는 변수 탐색 순서를 접안 위치, 입항 시간, 출항 시간, 크레인 관련 변수의 순으로 지정하되 각 범주 내에서는 희망 입항 시간이 빠른 선박부터 지정하도록 하였으며, 변수값은 방법 1과 마찬가지로 작은 값부터 할당하였다. 방법 3에서는 변수의 탐색 순서는 방법 2와 동일한 순서를 적용하였으며 변수값에 대해서는 접안 위치, 입항 시간, 출항 시간에 대해 선호 위치 또는 희망 입출항 시간으로부터 가까운 값을 우선적으로 할당하도록 하였다. 실험 결과의 항목으로는 최초해를 도출하기까지의 초 단위 수행 시간(Sec)과 목적함수 값(Obj)을 표시하였다.

실험 결과에 의하면 제약조건을 모두 만족하는 해가 도출되기까지 소요되는 시간은 큰 차이가 없는 것으로 나타났으나 해의 질 측면에 있어서는 큰 차이가 있음을 알 수 있다. 사실 방법 1과 방법 2는 모두 목적함수 측면을 전혀 고려하지 않은 방법이다. 그럼에도 불구하고 변수의 탐색 순서를 문제에 적합한 순서로 지정한 방법 2가 더 좋은 결과를 보임을 알 수 있다. 이는 문제의 특성에 따른 것으로 목적함수에 직접적인 영향을 미치는 접안 위치, 입항 시간, 출항 시간 중 특히 출항 시간과 관련된 것으로 파악된다. 즉, 방법 2에서는 출항 시간이 변수 순서상 비교적 빠른 순서로 값을 할당받음으로써 상대적으로 작은 값을 할당받을 수 있게 되어 목적함수가 크게 나빠지지 않고 있지만, 방법 1의 경우에는 크레인 관련 변수들에게 우선순위를 빼앗겨 작은 값이 할당되지 못하고 큰 값이 할당됨으로써 목적함수 값이 크게 나빠지게 된다. 방법 3은 목적함수를 명시적으로 고려한 것으로서 다른 두 가지 방법에 비해 목적함수 값이 월등히 좋아졌음을 확인할 수 있으며, 이를 통해 문제에 적합한

변수 탐색 순서 및 변수값 탐색 순서 지정 휴리스틱의 중요성을 알 수 있다.

표 1. 변수 및 변수값 지정 휴리스틱의 적용 결과
Table 1. Results of variable and value ordering

Data Set	선박 개수	방법1		방법2		방법3	
		Sec	Obj	Sec	Obj	Sec	Obj
1	15	3	2871	3	672	2	63
2	14	3	2900	2	679	2	8
3	13	2	1792	1	703	2	27
4	14	3	2365	2	822	2	12
5	13	3	2120	2	770	2	9
6	17	3	3953	3	1050	3	59
7	17	3	3498	2	1072	3	12
8	16	3	4101	3	1086	3	13
9	16	3	3834	2	770	4	20
10	20	3	5520	4	1275	4	28

표 2. 반복적 개선 탐색 방법의 적용 결과
Table 2. Results of iterative improvement search

Data Set	선박 개수	방법1		방법2		방법3		방법4	
		Sec	Obj	Sec	Obj	Sec	Obj	Sec	Obj
1	15	2	63	5	27	300	27	300	27
2	14	2	8	3	8	300	8	300	8
3	13	2	27	4	6	300	20	6	6
4	14	2	12	4	7	300	12	12	7
5	13	2	9	4	2	300	9	4	2
6	17	3	59	6	38	300	34	300	33
7	17	3	12	3	12	300	11	300	11
8	16	3	13	3	13	300	13	300	13
9	16	4	20	4	20	300	11	300	11
10	20	4	28	4	28	300	18	300	18
11	16	3	35	6	23	300	31	300	19
12	17	3	19	3	19	300	16	300	16
13	16	3	9	5	5	14	5	9	5
14	15	2	118	2	118	300	85	300	85
15	14	2	10	2	10	300	10	300	10
16	18	3	31	3	31	300	24	300	24
17	16	2	12	2	12	300	7	300	7
18	16	3	40	4	39	300	39	300	38
19	17	4	42	7	36	300	33	300	31
20	17	3	50	7	48	300	39	300	39
21	14	2	9	3	9	38	7	41	7
22	14	5	90	5	90	300	90	300	90
23	16	3	48	3	48	300	27	300	27
24	13	2	19	2	19	300	16	300	16
25	13	2	11	2	11	37	7	32	7

마지막 실험에서는 25개의 데이터 Set 각각에 대해 반복적 개선 탐색 수행 시 변수값 순서를 재정렬하는 방법과 목적함수 값을 제약조건으로 추가하는 방법을 각각 또는 결합하여 적용해 보았으며 실험 결과는 〈표 2〉와 같다. 방법 1은 〈표 1〉의 실험에서 방법 3과 동일하며 방법 2는 변수값 순서를 재정렬한 후 반복적으로 제약만족 탐색을 수행한 결과이다. 방법 3은 한 번의 제약만족 탐색만으로 이루어져 있으며 탐색 도중에 새로운 해가 도출될 때마다 그 해의 목적함수 값을 제약조건으로 추가하는 방법이다. 방법 4는 방법 2와 방법 3을 결합한 반복적 개선 탐색 방법이다. 음영처리된 결과는 각 데이터 Set에 있어서 가장 좋은 결과를 표시한 것이다. 각 실험에 대해 최대 수행 시간을 5분으로 제한하였다. 따라서 방법 3과 방법 4에서 5분이 경과하기 전에 프로그램이 중단된 경우의 최종해는 최적해를 의미하는 것이다.

〈표 2〉에서 보는 바와 같이 변수값 순서를 재정렬 하는 방법(방법 2)과 하나의 제약만족 탐색 내에서 목적함수를 새로운 제약조건으로 추가하는 방법(방법 3)을 각각 사용함으로써 대부분의 데이터 Set에 있어서 초기해보다 더 좋은 해를 비교적 빠른 시간 내에 도출해 낼 수 있음을 확인할 수 있다. 더욱이 반복적 개선 탐색의 틀 내에서 두 가지 방법을 모두 사용한 방법(방법 4)은 모든 데이터 Set에 대해 가장 좋은 해를 구할 수 있음을 알 수 있다.

Ⅵ. 결 론

선석 및 크레인 일정계획 문제는 복잡하고 다양한 제약조건을 포함하는 동시에 최적화 요소를 포함하고 있는 제약만족 및 최적화 문제이다. 본 논문에서는 이 문제의 해결을 위해 반복적 개선 탐색 과정의 틀 내에서 제약만족 탐색 기법을 적용하는 방안을 제시하였다. 반복적 개선 탐색 내에서 제약만족 탐색 수행 시 현재해의 목적함수 값을 이용하여 새로운 제약조건을 추가함으로써 현재해보다 좋은 해만을 대상으로 탐색을 수행할 수 있도록 하였으며, 이 방법과 변수값 탐색 순서를 재정렬하는 방법을 결합하여 적용하므로써 보다 빠르게 더 좋은 해가 도출될 수 있도록 하였다.

본 논문에서 제시한 방법은 반복적 개선 탐색 과정을 기본 틀로 하고 있기 때문에, 해상 운송의 특성상 선사측의 입출항 일정 변경 요청이 잦아 계획 자체가 수시로 변경 수립되어야 한다는 요구 조건에도 쉽게 부응하고 있다. 일정 변경 요청이 들어오면 그에 따라 접안 위치 변수의 순서 지정을 변경하여 제약만족 탐색을 재수행한 후, 필요 시 반복

적 개선 탐색 과정을 다시 적용하기만 하면 되기 때문이다. 또한 본 논문에서 제안한 방법은 일반적인 제약만족 및 최적화 문제의 해결을 위해 적용될 수 있을 것으로 판단되며, 이를 검증하기 위해 다양한 제약만족 및 최적화 문제로의 적용이 필요하다.

참고문헌

[1] 홍동희, 선수균, 이승명, 항만장비관리를 위한 멀티미디어 정보체제 구축 방안, 한국컴퓨터정보학회, 5권, 2호, 1-10, 2000.

[2] 홍동희, 이승명, 항만 게이트 자동화를 위한 최적 설계에 관한 연구, 한국컴퓨터정보학회, 6권, 2호, 58-64, 2001.

[3] Bartak, R., Constraint Programming : In Pursuit of the Holy Grail, Proceedings of WDS99(invited lecture), Prague, 1999.

[4] 류광렬, 김갑환, 백영수, 황준하, 박영만, 제약만족 탐색과 휴리스틱 교정기법을 이용한 최적 선석 및 크레인 일정계획, 한국지능정보시스템학회논문지, 6권, 2호, 1-14, 2000.

[5] Daganzo, C. F., The Crane Scheduling Problem, Transportation Research, 24B(3), 159-17, 1990.

[6] Peterkofsky, R. I. and Daganzo, C. F., A Branch and Bound Solution Method for the Crane Scheduling Problem, Transportation Research, 23B(3), 159-175, 1989.

[7] 윤철영, 문성혁, 컨테이너 터미널 사용자 비용을 최소로 하는 선석과 크레인의 최적 구성에 관한 연구, 한국항만학회지, 9권, 2호, 39-49, 1995.

[8] Park, K. T. and Kim, K. H.,. Berth Scheduling for Container Terminals by Using a Subgradient Optimization Technique, European Journal of Operational Research, 53, 1054-1064, 2002.

[9] Tsang, E., Foundations of Constraint Satisfaction, Academic Press Limited, 1996.

[10] Russell, S. and Norvig, P., Artificial Intelligence : A Modern Approach, Prentice Hall, 2003.

[11] Lawler, E. W. and Wood, D. E., Branch-and-bound methods: a survey, Operations Research, 14, 699-719, 1966.

[12] ILOG Solver, Reference and User Manual, Version 5.0, 2000.

저 자 소 개

황 준 하

1995년 부산대학교 컴퓨터공학과 졸업(공학사)

1997년 부산대학교 컴퓨터공학과 졸업(공학석사)

2002년 부산대학교 컴퓨터공학과 졸업(공학박사)

2002년 ~ 현재 금오공과대학교 컴퓨터공학부 교수

〈관심분야〉

인공지능, 최적화, 기계학습