韓國컴퓨터情報學會 論文誌
第14卷 第2號, 2009. 2.

2009-14-2-1-4

비선형 최적화 문제의 해결을 위한 정수계획법과 이웃해 탐색 기법의 결합

황 준 하*

Integration of Integer Programming and Neighborhood Search Algorithm for Solving a Nonlinear Optimization Problem

Junha Hwang*

요 약

정수계획법은 조합 최적화 문제의 최적해를 매우 효과적으로 탐색할 수 있는 기법인 반면에 대상 문제가 선형적으로 표현되어야만 적용이 가능하다는 단점이 있다. 본 논문에서는 정수계획법의 뛰어난 탐색 능력과 이웃해 탐색 기법의 유연성을 결합함으로써 비선형 최적화 문제를 효과적으로 해결하는 방안을 제시하고 있다. 먼저 1단계에서는 주어진 문제로부터 선형적으로 표현 가능한 부문제만을 대상으로 정수계획법을 적용한다. 2단계에서는 전체 문제를 대상으로 이웃해 탐색 기법을 적용하되 1단계의 결과를 초기해로 설정한 후 탐색을 수행한다. 비선형 최대 커버링 문제를 대상으로 한 실험 결과, 이와 같은 간단한 결합만으로도 이웃해 탐색 기법만을 적용했을 때보다 훨씬 좋은 해를 도출할 수 있음을 확인하였다. 이는 기본적으로 정수계획법의 탁월한 성능에 기인한 것으로 판단된다.

Abstract

Integer programming is a very effective technique for searching optimal solution of combinatorial optimization problems. However, its applicability is limited to linear models. In this paper, I propose an effective method for solving a nonlinear optimization problem by integrating the powerful search performance of integer programming and the flexibility of neighborhood search algorithms. In the first phase, integer programming is executed with subproblem which can be represented as a linear form from the given problem. In the second phase, a neighborhood search algorithm is executed with the whole problem by taking the result of the first phase as the initial solution. Through the experimental results using a nonlinear maximal covering problem, I confirmed that such a simple integration method can produce far better solutions than a neighborhood search algorithm alone. It is estimated that the success is primarily due to the powerful performance of integer programming.

▸ Keyword : 정수계획법(Integer Programming), 이웃해 탐색(Neighborhood Search), 최대 커버링 문제(Maximal Covering Problem)

• 제1저자 : 황준하
• 투고일 : 2009. 2. 5, 심사일 : 2009. 2. 11, 게재확정일 : 2009. 2. 20.
* 금오공과대학교 컴퓨터공학부 부교수
※ 본 논문은 금오공과대학교 학술연구비에 의하여 연구된 논문임.

Ⅰ. 서 론

인공지능(Artificial Intelligence) 분야와 경영과학(Operations Research) 분야는 각각 고유의 학문 분야로서 발전해 왔다. 인공지능 분야에서는 인간과 같이 사고하거나 행동하는 시스템을 개발하는 데 관심을 기울여 왔으며 최근에는 인간이 하는 일을 자동화하기 위한 보다 현실적인 알고리즘을 개발하기 위해 노력하고 있다. 경영과학 분야는 경영 문제에 대한 과학적 접근법을 개발하기 위한 학문 분야로서 2차 세계대전 당시 군사적 목적의 복잡한 문제를 해결하고자 한 것에 기원을 두고 있다.

인공지능과 경영과학 분야는 각각 다양한 이론을 토대로 여러 가지 문제를 해결하기 위해 노력해 왔다. 그 중에서 조합 최적화(Combinatorial Optimization) 문제는 전통적으로 두 가지 학문 분야 모두에 있어서 주된 공통 관심사이다. 조합 최적화 문제는 결정해야 될 변수들의 값이 정수인 최적화 문제의 일종으로서 특정 기준에 따른 변수 값들의 조합을 최대화하거나 최소화하는 문제로 표현되는데 이를 목적함수라고 부른다. 그 외에 부가적으로 변수들의 값에 대한 제약조건이 동반될 수 있다. 일반적으로 결정해야 할 변수들의 개수가 증가함에 따라 문제의 복잡도가 매우 높아지게 되고 최적해의 도출 또한 매우 어렵게 된다.

인공지능 분야에서는 조합 최적화 문제를 해결하기 위한 방안으로 탐색에 기반을 둔 다양한 휴리스틱 탐색 알고리즘들이 개발되어 왔다. 대표적인 알고리즘들로는 이웃해 탐색 기법의 일종인 언덕오르기 탐색(Hill-Climbing Search), 타부 탐색(Tabu Search)[1], 시뮬레이티드 어닐링(Simulated Annealing)[2]과 군집 탐색을 기반으로 하고 있는 유전 알고리즘(Genetic Algorithm)[3] 등이 있다. 이와 같은 알고리즘들은 대상 문제의 목적함수 또는 제약조건의 형태에 상관없이 다양한 종류의 조합 최적화 문제에 적용이 가능하다는 장점이 있으며, 지금까지의 연구 결과 많은 응용문제에 있어서 성능이 탁월한 것으로 알려져 있다. 그러나 거의 모든 알고리즘들에 있어서 그 성능에 대한 해석 방법이 수학적 지식을 기반으로 하기보다는 실험 결과와 직관적인 분석을 기반으로 하고 있다. 따라서 알고리즘의 행동에 대한 정확한 해석이 불가능하며, 도출해의 최적해 여부에 대한 판정 역시 불가능하다. 결국 이와 같은 알고리즘들은 최적해 도출을 목표로 하면서도 어느 정도 좋은 수준인 준최적해의 도출에 만족하게 된다.

반면에 경영과학 분야에서는 최적해의 도출을 보장함과 동시에 효율적으로 최적해를 도출할 수 있는 방안이 개발되어 왔는데, 이른바 정수계획법(Integer Programming)으로 알려져 있다[4]. 이 방법이 최적해의 도출을 보장하는 이유는 기본적으로 가능한 후보해들을 모두 다 나열하기 때문이다. 그러면서도 높은 효율을 자랑하는 것은 수학적 해법을 토대로 하는 선형계획법(Linear Programming)을 적용하고 있기 때문이다. 그러나 선형계획법을 적용하기 위해서는 문제의 목적함수와 제약조건이 선형식으로 표현되어야만 한다는 제한이 따른다. 즉, 그만큼 정수계획법의 표현력이 떨어지게 되며 적용 가능한 문제 역시 좁아지게 된다. 따라서 현재는 경영과학 분야에서도 비선형 문제의 해결을 위해 휴리스틱 탐색 알고리즘들을 동원하고 있다.

지금까지 동일한 대상 문제를 해결하기 위해 휴리스틱 탐색 알고리즘과 정수계획법은 별도로 적용되어 왔으며 상호 비교를 통해 해당 알고리즘의 우수성을 입증해 왔다. 물론 대상 문제 자체가 선형식으로 표현될 수 없는 경우에는 정수계획법의 적용 자체가 불가능하다. 그러나 본 논문에서는 하나의 비선형 문제를 해결하기 위해 휴리스틱 탐색 알고리즘들 중 이웃해 탐색 기법과 정수계획법을 효과적으로 결합하는 방안을 제시하고 있다. 우선 대상 문제의 특성을 분석하여 선형식으로 표현 가능한 부분만을 분리해 낸다. 그리고 먼저 선형식으로 표현 가능한 문제만을 대상으로 정수계획법을 적용하여 주어진 시간 동안 최적해 또는 준최적해를 도출한다. 마지막으로 정수계획법을 통해 도출한 해를 초기해로 하여 이웃해 탐색 기법을 수행하되 선형식으로의 표현이 불가능한 부분을 포함한 문제 전체를 대상으로 수행하게 된다.

본 연구에서는 비선형 최대 커버링 문제(Maximal Covering Problem)를 대상으로 제안한 기법의 유용성을 검증하였다. 실험 결과 선형식으로 표현 가능한 부분만을 대상으로 했을 때 정수계획법은 다른 이웃해 탐색 기법들에 비해 월등한 성능을 발휘하였다. 뿐만 아니라 이웃해 탐색 기법들과는 달리 최적해의 도출 여부도 확인할 수 있었다. 또한 정수계획법으로부터 도출된 해를 초기해로 설정한 후 전체 문제에 대해 이웃해 탐색 기법을 수행한 결과 처음부터 이웃해 탐색 기법만을 수행했을 때보다 훨씬 더 좋은 해를 도출할 수 있었다. 이것 역시 정수계획법에 의한 초기해의 우수성에 기인한 것으로 판단된다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 본 논문의 대상 문제인 최대 커버링 문제에 대해 설명하고 아울러 관련 연구에 대해 소개한다. Ⅲ장에서는 정수계획법과 이웃해 탐색 기법의 기본 알고리즘 및 대상 문제로의 적용 방안을 각각 설명한 후 두 가지 기법을 결합하는 방안에 대해 기술한다. Ⅳ

장에서는 각 기법들에 대한 실험 결과를 분석하며 마지막으로 V장에서 결론 및 향후 과제를 설명한다.

II. 대상 문제 및 관련 연구

1. 대상 문제

본 논문은 최대 커버링 문제(MCP)를 대상으로 한다. MCP는 0과 1로 이루어진 mxn 행렬이 주어진 경우 n개의 열로부터 주어진 d개의 열을 선택하되 가장 많은 행을 커버하는 문제로 정의된다[5]. 〈그림 1〉은 5행 4열로 이루어진 MCP의 예로서 각 열마다 서로 다른 행을 커버하고 있다. 여기서 선택해야 할 열의 개수 d의 값이 2로 주어진다면 가장 많은 행을 커버하는 경우는 열 1과 4를 선택하는 것이다.

	1	2	3	4
1	1	1		
2		1	1	1
3				
4	1		1	
5				1

그림 1. 최대 커버링 문제의 예
Fig. 1. An example of maximal covering problem

MCP는 식 (1)과 같이 표현될 수 있다. 단, 이 수식은 앞서 MCP를 가장 많은 행을 커버하는 문제로 정의한 것과는 달리 커버되지 못하는 행(이를 공백행이라 부른다)을 최소화하는 문제로 표현한 것이다. 공백행을 최소화하는 것은 가장 많은 행을 커버하는 것과 동일한 의미로서 MCP의 정의와 부합하는 것이다.

식 (1)에서 목적함수는 m개의 행들 중 공백행을 최소화함을 의미하며, 첫 번째와 두 번째 제약조건은 n개의 열들 중 d개를 선택함을 의미한다. 세 번째 제약조건에서 a_{ij}는 주어진 행렬의 i행 j열의 상수값을 나타내는 것으로서 세 번째와 네 번째 제약조건은 만약 i번째 행이 공백행이라면 y_i의 값이 1이 되게 만들고 한 개 이상의 열에 의해 커버된다면 0이 되도록 만든다. 바로 세 번째와 네 번째 제약조건들에 의해 목적함수가 본래의 목적인 공백행의 최소화를 달성할 수 있게 된다.

$$\text{Minimize} \quad \sum_{i=1}^{m} y_i$$

$$\text{Subject to} \quad \sum_{j=1}^{n} x_j = d$$

$$x_j \in \{0,1\},\ j=1,\ldots,n \quad \cdots\cdots (1)$$

$$\sum_{j=1}^{n} a_{ij} x_j + y_i \geq 1$$

$$y_i \in \{0,1\},\ i=1,\ldots,m$$

이와 같이 기본적으로 MCP는 선형식으로 표현될 수 있다. 그러나 MCP로 표현되는 실세계 문제들의 경우 선형식으로 표현될 수 없거나 선형식으로 표현된다 하더라도 현실적으로 정수계획법의 적용이 불가능한 요소를 포함하는 경우가 많다. MCP로 표현되는 승무일정계획 문제의 예를 들어보자. 이 문제의 경우 하나의 열은 한 승무원에 대한 출근부터 퇴근까지의 열차 운행 경로를 의미하며, 이로부터 해당 승무원의 출근시간과 퇴근시간이 결정된다. 출퇴근시간과 관련하여 선택된 d개의 열들은 출근시간과 퇴근시간이 서로 엇갈리는 경우를 선호하지 않는다. 즉, 먼저 출근한 승무원이 먼저 퇴근하는 것을 더 선호하며 본 논문에서는 이를 출퇴근 선호조건이라 부른다. 출퇴근 선호조건을 선형식으로 표현하는 것이 불가능한 것은 아니다. 모든 열의 쌍들 중 출근시간과 퇴근시간이 엇갈리는 쌍이 동시에 선택되는 경우를 최소화하면 되며 이를 수식으로 표현하면 식 (2)와 같다[6]. V는 출발시간과 도착시간이 엇갈리는 쌍들의 집합이고 z_{ij}는 열 i와 열 j가 동시에 선택된 경우 1이 되는 결정변수로서 V의 원소이다.

$$\text{Minimize} \sum_{(i,j)\in V} z_{ij}$$

$$\text{Subject to} \quad x_i + x_j - 1 \leq z_{ij},\ (i,j) \in V \quad \cdots\cdots (2)$$

$$x_j \in \{0,1\},\ i=1,\ldots,n$$

$$z_{ij} \in \{0,1\},\ (i,j) \in V$$

그러나, 선형식으로 표현하는 것이 가능하다고 하여 정수계획법의 적용이 가능한 것은 아니다. 이 경우 결정 변수의 개수가 너무 많아지기 때문이다. 예를 들어 열의 개수가 10,000개라고 가정하면 최악의 경우 1억 개라는 매우 큰 수의 변수를 필요로 하게 된다. 실제로 본 연구에서 사용한 데이터들 중 규모가 가장 작은 데이터만 하더라도 메모리 문제로 인해 정수계획법의 적용이 불가능함을 실험을 통해 확인하였다.

2. 관련 연구

인공지능 분야와 경영과학 분야에서는 대상 문제의 유사성 및 최적화라는 목적의 유사성으로 인해 서로의 기법을 상호 보완적으로 활용하기 위한 연구가 활발히 진행되어 왔다. 대표적인 예가 제약만족기법(Constraint Satisfaction Technique)과 정수계획법의 결합이다. 제약만족기법은 인공지능 분야에서 개발된 탐색 알고리즘으로서 주로 복잡한 제약조건을 포함하는 문제를 해결하는 데 적용되고 있다[7]. 제약만족기법과 정수계획법은 전통적으로 승무일정계획을 수립하기 위해 적용되었는데 여기서는 하나의 문제를 해결하기 위해 두 가지 기법이 적용되었다기보다는 하나의 큰 문제 내에 포함된 부문제들을 각자 해결함으로써 서로 보완적인 역할을 담당하였다[8]. 최근에는 제약만족기법과 정수계획법을 보다 밀접하게 결합하기 위한 연구도 진행되고 있다[9].

MCP를 해결하기 위한 종래의 연구로는 정수계획법의 효율을 향상시키기 위한 연구가 많이 진행되었다[10]. 이 방법은 최적해의 도출을 보장하지만 문제 자체가 선형식으로 표현되는 경우만을 대상으로 하고 있어 표현력에 한계가 있다. 보다 최근에는 휴리스틱 탐색 기법들을 활용하는 경우도 있었다. [11]에서는 유전 알고리즘을 적용하되 비발현 유전자의 개념을 도입하여 보다 효과적으로 탐색을 수행할 수 있도록 하였다. 또한 [12]에서는 단순 언덕오르기 탐색(Simple Hill-Climbing Search)과 시뮬레이티드 어닐링이 다른 이웃해 탐색 기법에 비해 효과적임을 확인하였다. 그러나 이 연구들에서는 휴리스틱 탐색 기법만을 대상으로 하고 있기 때문에 정수계획법과의 비교 연구가 부족하였다.

MCP를 위한 휴리스틱 탐색 기법과 정수계획법의 결합에 관한 연구는 비교적 드문 편이지만 몇몇 연구를 통해 이에 대한 연구가 진행되었다. [13]에서는 승무일정계획 수립을 위해 정수계획법과 휴리스틱 탐색기법을 결합하여 사용하였다. 하지만 MCP 자체를 풀기 위해서는 정수계획법만 적용되었고 그 결과를 개선하기 위한 방안으로 휴리스틱 탐색 기법이 적용되었다. 결국 서로 별개의 문제를 풀기 위해 적용되었다고 할 수 있다. [6]에서는 정수계획법과 이웃해 탐색 기법을 보다 긴밀하게 결합하였는데, 이웃해 탐색 기법을 주로 하여 탐색 다각화를 위해 정수계획법을 활용하였다. 그러나 하나의 데이터만을 대상으로 하였기 때문에 정수계획법의 효과를 제대로 분석하기 어려웠던 것으로 사료된다.

본 논문에서는 우선 선형 요소만을 포함하는 MCP에 있어서 정수계획법과 이웃해 탐색 기법의 성능을 비교하였으며, 이를 바탕으로 정수계획법과 이웃해 탐색 기법을 적절히 결합함으로써 비선형 요소가 포함된 MCP에 대해서도 효과적으로 문제를 해결할 수 있는 방안을 제시하고 있다.

III. 정수계획법과 이웃해 탐색 기법의 적용

1. 정수계획법의 적용

정수계획법은 선형계획법을 기반으로 하고 있다. 정수계획법 문제에서 결정변수들이 정수라는 제약조건을 완화하여 실수로 확대하는 경우 바로 선형계획법 문제가 된다. 선형계획법은 기본적으로 단체법(simplex method)이라는 해법을 사용하고 있는데 이 방법은 1947년 Dantzig에 의해 제시된 이래 지금까지도 선형계획 문제를 풀기 위한 가장 뛰어난 방법으로 인정받고 있다[14]. 단체법에 의하면 수리적 모델을 통해 최적해의 가능성이 있는 몇몇 후보해들만을 대상으로 최적여부를 판단함으로써 매우 빠르게 최적해를 도출할 수 있게 된다. 단, 이는 문제 자체가 선형식으로 표현되어야만 가능한 것이다.

정수계획법을 위해 사용되는 대표적인 해법으로는 분지한계법(branch and bound)이 있다[4]. 분지한계법은 1960년 Land와 Doig에 의해 제안된 후 Dakin 등에 의해 발전되어 왔다. 이 방법은 모든 가능해를 열거하는 방식의 일종이지만 해를 분지하면서 상한과 하한을 사용하여 해의 타당성을 빠르게 판별함으로써 해의 열거를 최소화하는 방식이다. 이때 상한 또는 하한값을 구하기 위해 정수 제약조건을 완화하여 선형계획법 문제를 풀게 된다.

본 논문에서는 정수계획법을 구현하기 위해 ILOG CPLEX를 사용하였다[15]. ILOG CPLEX는 선형계획법 및 정수계획법을 구현하기 위한 상용 도구로서 현재 가장 효율적인 정수계획법 개발 도구로 알려져 있다. 정수계획법을 MCP에 적용하는 경우에는 II장 1절의 식 (1)과 같이 선형식으로 표현 가능한 부분만을 대상으로 한다.

2. 이웃해 탐색 기법의 적용

이웃해 탐색 기법은 하나의 완전한 해로부터 출발하여 이웃해를 반복적으로 탐색함으로써 해를 개선시켜 나가는 방식으로서 언덕오르기 탐색, 단순 언덕오르기 탐색(Simple Hill-Climbing Search), 시뮬레이티드 어닐링, 타부 탐색 등이 있다. 언덕오르기 탐색과 타부 탐색은 모든 이웃해들 중 가장 좋은 해를 다음 해로 선택하는 최급 상승(steepest

ascent) 전략을 기반으로 하고 있는 반면에 단순 언덕오르기 탐색과 시뮬레이티드 어닐링은 현재해로부터 무작위로 하나의 이웃해를 생성한 후 현재해보다 좋으면 이동하는 최우선 향상(first improvement) 전략을 기반으로 하고 있다. 기존 연구 [12]에 의하면 이웃해 탐색 기법들 중에서는 최우선 향상 전략을 기반으로 하고 있는 단순 언덕오르기 탐색과 시뮬레이티드 어닐링의 성능이 더 좋은 것으로 확인되었다. 따라서 본 연구에서도 정수계획법과의 비교 및 결합을 위해 이웃해 탐색 기법들 중 단순 언덕오르기 탐색과 시뮬레이티드 어닐링만을 고려하였다.

MCP를 위한 단순 언덕오르기 탐색 알고리즘은 〈그림 2〉와 같이 [12]에서의 적용 방법과 동일하다. 먼저 초기해 x를 생성하기 위해 그리디 추가(greedy adding) 휴리스틱을 사용하였다. 즉, 열을 하나씩 선택하되 현재 가장 많은 공백행을 커버할 수 있는 열을 선택하며, 이와 같은 과정을 지정한 열 선택 개수인 d만큼 반복 수행한다. 이웃해 생성 방법으로는 k-exchange를 사용하였다. k-exchange는 이웃해 하나를 생성할 때 k개 이하의 열을 교체하는 것을 의미한다. 예를 들어 5-exchange의 경우에는 1개 이상 5개 이하의 열을 교체하여 이웃해를 생성하게 되는데 실제 교체되는 열의 개수는 매번마다 무작위로 결정된다. 〈그림 2〉에서는 k-exchange에 의해 생성될 수 있는 이웃해의 집합을 $S_k(x)$로 표현하였다. 실험적으로는 k의 값이 3 이상 5 이하일 때 비교적 좋은 성능을 발휘하는 것으로 확인되었다. 교체해야 할 열의 개수가 결정되면 제거와 추가가 순차적으로 수행된다. 제거 시에는 대상 열들이 무작위로 선택되며 추가 시에는 초기해 생성 시와 마찬가지로 그리디 추가 휴리스틱을 사용한다. 하나의 이웃해가 생성되면 현재해와 비교하여 목적함수 값이 더 좋거나 같다면 이웃해를 현재해로 변경한 후 탐색을 계속 진행하며, 더 좋지 않다면 현재해를 기반으로 또 다시 이웃해를 조사하게 된다. 이상의 과정을 일정 시간 동안 반복 수행한다.

시뮬레이티드 어닐링 역시 단순 언덕오르기 탐색과 매우 유사하다. 단순 언덕오르기 탐색과 마찬가지로 이웃해 하나를 생성한 후 현재해보다 더 좋거나 같다면 이웃해를 현재해로 설정한 후 탐색을 진행해 나간다. 그렇지만 단순 언덕오르기 탐색과는 달리 현재해보다 좋지 않다 하더라도 이동이 가능하다. 이때 이동 확률은 $e^{-\Delta E/t}$와 같이 결정되는데 ΔE는 해의 개선 정도인 $(Cost(x^*) - Cost(x))$를 의미한다. 즉, 좋지 않은 정도가 커지면 커질수록 이동 가능성은 낮아지게 된다. 그리고 온도를 의미하는 t는 탐색 초기에는 높은 값을 가지다가 탐색이 진행될수록 점점 감소하게 된다. 이로 인해 탐색 초기에는 좋지 않은 해로의 이동 확률이 높은 반면에 탐색이 진행될수록 이동 확률이 낮아지게 되며 결국 t 값이 0이 되면 단순 언덕오르기 탐색과 같아진다.

```
Algorithm k-exchange_Simple_Hill_Climbing_Search
    Ω : Set of candidate solutions.
    x : Current solution.
    Cost : Objective function.
    ℵ(x) : Neighborhood of x ⊂ Ω.
    S_k(x) : k-exchange neighborhood ⊂ ℵ(x).
Begin
    Start with an initial solution x ∈ Ω
    While stopping condition is not met Do
        Generate a neighbor solution x* ∈ S_k(x)
        If Cost(x*) ≤ Cost(x) Then
            x = x*
    End While
    return x
End Begin
```

그림 2. MCP를 위한 단순 언덕오르기 탐색
Fig. 2. Simple Hill-Climbing Search for MCP

3. 정수계획법과 이웃해 탐색 기법의 결합

본 논문에서 제안하는 정수계획법과 이웃해 탐색 기법의 결합 방법은 매우 간단한다. 〈그림 3〉은 정수계획법과 단순 언덕오르기 탐색을 결합한 알고리즘을 보인 것이다. 여기서 보는 바와 같이 1단계에서는 정수계획법을 수행하며 2단계에서는 1단계에서 도출한 해를 초기해로 하여 단순 언덕오르기 탐색을 수행한다.

이때 각 단계 별로 후보해 x를 평가하는 목적함수 $Cost(x)$가 달라지게 된다. 1단계에서 정수계획법이 적용되는 경우에는 목적함수와 제약조건이 모두 선형식으로 표현 가능해야 하므로 공백행의 개수가 목적함수가 되며 이를 최소화하는 것이 1단계의 목표이다. 2단계에서 단순 언덕오르기 탐색이 수행되는 경우에는 어떤 형태의 목적함수 및 제약조건도 표현이 가능하다. 따라서 (공백행의 개수 + α × 출퇴근 선호조건 위배 개수)로 표현된다. 여기서 α는 공백행의 개수와 출퇴근 선호조건 사이의 상대적 중요도를 나타내는 상수이다. 이 값이 커지게 되면 탐색 알고리즘은 출퇴근시간 제약조건을 위배하는 경우를 최소화하기 위해 더 많은 노력을 기울이게 되며, 이 값이 작아지게 되면 공백행의 개수를 최소화하는 데 더 많은 노력을 기울이게 된다. 결국 1단계에서는 공백행을 최소화하는 데 전력을 기울이게 되며 2단계에서는 1단계에서의 결과를 바

탕으로 공백행의 최소화를 양보하면서 출퇴근시간 선호조건을 포함한 전체 목적함수 측면에서의 최적해를 탐색해 나간다. 2단계에서 단순 언덕오르기 탐색 대신 시뮬레이티드 어닐링이 적용되는 경우에도 전체적인 알고리즘은 동일하다.

Algorithm Integration of Integer Programming and Simple Hill-Climbing Search
Phase 1. Integer Programming
Let $Cost(x) = \sum_{i=1}^{m} y_i$
y = Minimize $Cost(x)$
Phase 2. Simple Hill-Climbing Search
Let $Cost(x) = \sum_{i=1}^{m} y_i + \alpha \sum_{(i,j)\in V} z_{ij}$
Start with the initial solution y from Phase 1.
While stopping condition is not met **Do**
Generate a neighbor solution $x^* \in S_k(x)$
If $Cost(x^*) \leq Cost(x)$ **Then**
$x = x^*$
End While
return x
End Begin

그림 3. 정수계획법과 단순 언덕오르기 탐색의 결합
Fig. 3. Integration of Integer Programming and Simple Hill-Climbing Search

정수계획법과 이웃해 탐색 기법을 결합한 방식이 단순히 이웃해 탐색 기법만을 적용했을 때보다 더 좋은 성능을 발휘할 수 있느냐 하는 것은 1단계에서의 정수계획법의 성능에 좌우된다고 할 수 있다. 수리적 모델을 기반으로 하고 있는 정수계획법의 특성 상 1단계의 대상 문제와 같이 정수계획법을 적용할 수 있는 문제에 있어서는 최적해를 탐색하는 데 정수계획법이 이웃해 탐색 기법보다 효과적일 것으로 기대된다. 또한 1단계에서 정수계획법에 의해 더 좋은 해가 도출될 수 있다면 2단계에서의 전체적인 목적함수 측면에서도 보다 쉽게 더 좋은 해를 도출할 수 있을 것으로 기대된다.

IV. 실험 결과

실험을 위해 사용된 데이터는 〈표 1〉과 같다. MCP814는 국내 모 도시의 지하철 일일 승무일정계획 데이터로서 MCP로 모델링할 경우 각 행은 승무원이 운행할 최소 단위의 열차 운행 구간을 의미하며, 각 열은 한 승무원이 하루 동안 담당할 열차 운행 구간들의 집합으로 일일 근무계획에 해당된다. 또한 각 승무원의 출근시간과 퇴근시간이 기록되어 있다. MCP814의 경우 m = 814, n = 179,514, d = 83으로서 약 18만개의 열들 중 83개를 선택하여 814개의 행을 최대한 많이 커버하는 문제이다. MCP634를 비롯한 다른 데이터들은 실험을 위해 다양한 규모로 인공적으로 생성한 데이터이다. 모든 실험은 Pentium IV 3.4GHz, 1G RAM PC 상에서 수행되었으며 정수계획법 구현 도구로는 ILOG CPLEX 10.2를 사용하였다[15].

표 1. 실험 데이터
Table 1. Experimental data

특징 / data	행의 수(m)	열의 수(n)	선택 열의 수(d)	각 열이 커버하는 행의 수
MCP312	312	14,326	33	7, 8
MCP384	384	40,544	40	8, 9
MCP453	453	72,094	45	9, 10
MCP520	520	111,578	53	9, 10
MCP634	634	142,265	65	10
MCP814	814	179,514	83	10

실험은 크게 두 가지로 나누어진다. 첫 번째는 선형식으로 표현 가능한 공백행 최소화 문제만을 대상으로 하여 이웃해 탐색 기법들과 정수계획법의 성능을 비교하는 것이며, 두 번째는 출퇴근 선호조건을 포함한 전체 문제를 대상으로 이웃해 탐색 기법만을 적용했을 때와 이웃해 탐색 기법과 정수계획법을 결합한 기법을 적용했을 때의 성능을 비교하는 것이다.

첫 번째, 선형식으로 표현 가능한 공백 최소화 문제를 대상으로 정수계획법(IP)과 이웃해 탐색 기법인 단순 언덕오르기 탐색(SHC) 및 시뮬레이티드 어닐링(SA)을 각각 적용했을 때의 실험 결과는 〈표 2〉와 같다. SHC와 SA는 각 실험당 1시간의 시간 제한을 두고 수행하였으며 실험 결과 수치는 각각 5회 실험 후 평균값을 나타낸 것이다. SHC와 SA의 이웃해 생성 방법으로는 5-exchange를 사용하였다. IP의 경우에는 1시간의 시간 제한을 두고 수행하되 그 전에 최적해가 도출되면 수행을 중단하였으며 그 때까지의 소요 시간을 기록하였다. 그리고 IP의 경우에는 매 실험마다 동일한 결과를 보이므로 단 한 번의 실험 결과만을 나타낸 것이다.

〈표 2〉에 의하면 SA가 SHC보다는 약간 좋은 성능을 발휘하고 있는 것으로 보이나 그 차이는 미미한 편이다. 그런데 놀랍게도 IP는 SHC와 SA에 비해 월등히 좋은 결과를 보였다.

더군다나 상대적으로 규모가 작은 데이터인 MCP312, MCP384, MCP453, MCP520에 대해서는 매우 짧은 시간 내에 최적해를 도출할 수 있었으며, MCP634와 MCP814에 대해서도 최적해라고 보장할 수는 없지만 최적해로 추정되는 해를 도출할 수 있었다. 정수계획법은 선형식으로 표현된 문제에 대하여 최적해를 효율적으로 도출할 수 있을 뿐만 아니라 최적해의 도출 여부도 판정할 수 있다는 특성을 가지고 있다. 본 실험은 이와 같은 정수계획법의 특성을 잘 보여주고 있다.

두 번째, 공백행 최소화와 출퇴근 선호조건을 모두 고려한 실험에서의 적용 기법은 총 4가지이다. 첫 번째는 단순 언덕오르기 탐색(SHC)만을 적용하는 것이고 두 번째는 시뮬레이티드 어닐링(SA)만을 적용하는 것이다. 세 번째는 정수계획법과 단순 언덕오르기 탐색을 결합(IP+SHC)하여 적용하는 것이며 네 번째는 정수계획법과 시뮬레이티드 어닐링을 결합(IP+SA)하여 적용하는 것이다.

표 2. 공백행 최소화 실험 결과
Table 2. Experimental results for minimizing uncovered rows

기법 / data	SHC	SA	IP		
			공백행	소요 시간(초)	최적해 보장 여부
MCP312	4.6	3.8	2	77	O
MCP384	5	4	1	160	O
MCP453	11.8	11.8	8	575	O
MCP520	4.4	4.8	2	1116	O
MCP634	5	4.2	1	-	X
MCP814	7.4	6.4	1	-	X

본 실험을 위해서는 두 가지 파라미터를 결정해야 한다. 첫 번째는 〈그림 3〉의 알고리즘 2단계에서 밝힌 바와 같이 목적함수 내에서 공백행 최소화와 출퇴근시간 선호조건 위배 최소화를 결합하는 방식으로서 α 값을 결정해야 한다. 본 연구에서는 α 값을 0.5로 설정하였다. 이는 실험적으로 결정된 값으로서 가급적이면 공백행의 최소화에 영향을 주지 않으면서도 출퇴근시간 제약조건을 위배하는 경우를 최소화할 수 있는 값으로 설정하였다. 그러나 이 값이 특별한 의미를 갖는 것은 아니며 공정한 비교를 위해 모든 기법에 있어서 동일한 값이 적용되기만 하면 된다. 어떤 값이 오더라도 전체적인 결과의 경향에는 영향을 미치지 않기 때문이다. 두 번째는 1단계와 2단계의 수행 시간을 결정해야 한다. SHC 또는 SA만을 적용할 경우에는 처음부터 전체 문제에 대한 최적화를 수행하기 때문에 이에 대한 결정이 불필요하지만, IP+SHC와 IP+SA의 경우에는 1단계와 2단계에서 각 기법이 수행되는 시간을 결정할 필요가 있다. 본 실험에서는 전체 수행 시간의 70%동안 IP를 수행하였으며 나머지는 이웃해 탐색 기법을 수행하였다. 물론 IP에게 주어진 수행 시간이 끝나기 전에 최적해가 도출된다면 그 이후부터 나머지 시간동안 이웃해 탐색 기법이 수행된다. 본 실험에서도 첫 번째 실험과 마찬가지로 각 실험 당 1시간의 시간 제한을 두었으며 최종 실험 결과는 5회 실험 후 평균값을 나타내었다.

두 번째 실험 결과는 〈표 3〉과 같다. 여기서는 최종 결과에 대한 공백행의 개수와 출퇴근시간 선호조건의 위배 개수 그리고 최종 목적함수 값을 함께 표시하였다. 여기서 중요한 수치는 최종 목적함수 값이다. 공백행의 개수와 출퇴근시간 제약조건의 위배 개수는 참고 사항으로서 대부분 출퇴근시간 제약조건을 위배하는 경우가 드물게 나타남을 알 수 있다. 〈표 3〉에서 우선 SHC와 SA의 목적함수 값을 비교해 보면 MCP384를 제외한 모든 데이터들에 있어서 SA가 SHC에 비해 다소 우세하게 나타나고 있다. 새로이 고려해야 될 목적함수가 추가된 후에도 여전히 SHC와 같이 계속해서 더 좋은 해만을 고려했을 때보다는 SA와 같이 더 좋지 않은 방향으로의 이동을 허용함으로써 더 좋은 해로 이동할 수 있는 기회가 높아지는 것으로 판단된다. 그러나 여기서도 더 중요한 사실은 IP+SHC와 IP+SA가 SHC나 SA에 비해 월등히 좋다는 것이다. 이것은 1단계에서의 IP의 뛰어난 성능에 기인한 것으로 판단된다. 즉, 비록 결합 방법의 2단계에서 SHC나 SA의 수행 시간이 각각 단독으로 수행했을 때보다 훨씬 짧지만 1단계에서 IP에 의해 도출된 매우 좋은 해를 초기해로 설정한 후 탐색을 수행했기 때문인 것으로 분석된다. 한편 IP+SHC와 IP+SA에 있어서는 거의 유사한 성능을 발휘하고 있는 것으로 나타났다. 2단계에서 SHC나 SA 중 어떤 이웃해 탐색 기법을 사용하느냐는 전체적인 성능에 크게 영향을 미치지 않는 것으로 판단된다.

표 3. 전체 목적함수에 대한 실험 결과
Table 3. Experimental results of the total objective function

data / 기법		SHC	SA	IP+SHC	IP+SA
MCP312	공백행	6	4.6	2.4	2.4
	출퇴근위배	0	0	0	0
	목적함수	6	4.6	2.4	2.4

MCP384	공백행	7.2	7.2	1.6	1.4
	출퇴근위배	0	0	0.6	0.6
	목적함수	7.2	7.2	1.9	1.7
MCP453	공백행	16	14.8	10	10.2
	출퇴근위배	0.2	0.2	0	0.6
	목적함수	16.1	14.9	10	10.5
MCP520	공백행	10.4	8.8	4.2	4.6
	출퇴근위배	0.2	0	0.6	0
	목적함수	10.5	8.8	4.5	4.6
MCP634	공백행	14.4	12.8	6.2	5.6
	출퇴근위배	0.6	0.8	1	2
	목적함수	14.7	13.2	6.7	6.6
MCP814	공백행	15.4	13.6	8.0	6.8
	출퇴근위배	0.6	1.2	2.8	3.4
	목적함수	15.7	14.2	9.4	8.5

〈그림 4〉는 두 번째 실험에서 SHC, SA, IP+SHC 각각을 통한 1회 실험 시 수행시간에 따른 목적함수 값의 변화를 나타낸 것으로서 각 기법에 대한 전형적인 변화 패턴을 보여주고 있다. 참고로 IP+SA는 IP+SHC와 유사한 패턴을 나타낸다.

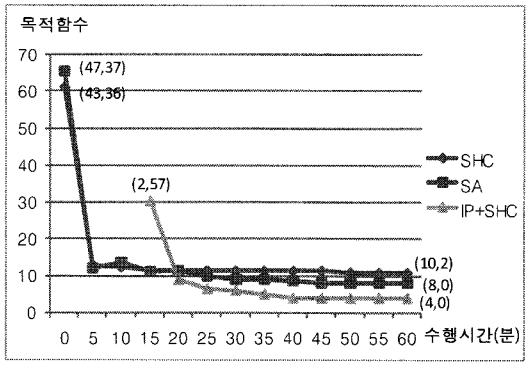

그림 4. SHC, SA, IP+SHC의 탐색 추이
Fig. 4. Search patterns of SHC, SA, IP+SHC

실험 데이터로는 MCP520을 사용하였으며 추세선 내부의 숫자는 해당 시점에서의 (공백행의 개수, 출퇴근 선호조건 위배 개수)를 의미한다. SHC와 SA 모두 탐색 초반에 매우 빠른 속도로 해가 개선되고 있다. IP+SHC의 경우에는 15분 경에 IP 탐색 결과 공백행 최소화 측면의 최적해를 도출한 후 이후로는 전체 목적함수에 대한 최소화를 진행하게 되는데 이 때 역시 탐색 초반에 매우 빠른 속도로 해가 개선되고 있다. 1단계에서의 IP를 통해 공백행의 개수 측면에서 보다 더 좋은 해로부터 출발함에 따라 최종적으로 더 좋은 해를 도출할 수 있음을 알 수 있다. 이 결과가 어떤 경우라도 IP+SHC나 IP+SA가 SHC나 SA보다 유용함을 나타내는 것은 아니다. 예를 들어 아주 짧은 시간 내에 어느 정도 좋은 해를 요구하는 응용문제의 경우에는 SHC 또는 SA가 보다 유용할 수도 있다.

V. 결론 및 향후 과제

정수계획법과 이웃해 탐색 기법은 모두 조합 최적화 문제를 해결하기 위해 적용되어 왔지만 알고리즘의 특성으로 인해 별도로 적용되는 경우가 대부분이었다. 정수계획법은 수리적 모델을 기반으로 최적해를 효율적으로 탐색할 수 있는 기법이지만 문제가 선형식으로 표현되어야 한다는 단점이 있다. 반면에 이웃해 탐색 기법은 표현력은 강하지만 탐색 방향에 대한 해석이 어려우며 최적해의 도출을 보장할 수 없다는 단점이 있다. 본 논문에서는 정수계획법과 이웃해 탐색 기법을 결합함으로써 비선형 최적화 문제를 보다 효과적으로 해결할 수 있는 방안을 제시하였다. 이를 위해 먼저 대상 문제인 최대 커버링 문제로부터 선형식으로 표현 가능한 부분을 도출한 후, 1단계에서는 선형 문제만을 대상으로 정수계획법을 적용함으로써 빠른 시간 내에 최적해를 도출할 수 있도록 하였으며 2단계에서는 1단계의 결과를 바탕으로 전체 비선형 문제를 해결하도록 하였다. 실험 결과 본 논문에서 제시한 기법은 이웃해 탐색 기법만을 적용할 때보다 훨씬 좋은 해를 도출할 수 있음을 확인하였다.

향후 연구 과제로는 정수계획법에 대한 추가 실험 및 분석이 뒤따라야 할 것으로 사료된다. 본 논문의 실험 결과에서 알 수 있듯이 정수계획법의 성능은 매우 뛰어나다. 그러나 이론적 배경을 충분히 이해하지 않고서는 정수계획법을 보다 세밀하게 다루기가 어려워진다. 예를 들어 본 논문에서는 최대 커버링 문제를 최대화 문제에서 공백행을 최소화하는 최소화 문제로 변경하여 적용하였다. 의미는 동일하지만 각각에 대한 수식이 달라지게 되며 ILOG CPLEX를 적용한 결과 최적해를 도출하기까지 소요 시간에 있어서 큰 차이를 보임을 확인하였다. 데이터에 따라 다르지만 대부분 최대화 문제로 모델링한 경우 소요 시간이 크게 증가하는 것으로 나타났다. 이에 대한 해석을 위해서는 경영과학 측면에서의 분석이 필요하지

만, 이웃해 탐색 기법과 같은 휴리스틱 탐색 기법으로부터 출발한 본 연구의 특성상 아직까지 이에 대한 해석이 불충분하다. 따라서 향후로 정수계획법의 이론적 배경을 바탕으로 이에 대한 분석이 필요하다. 아울러 이웃해 탐색 기법과 정수계획법의 비교 연구를 통해 선형적 문제에 대한 이웃해 탐색 기법의 효율을 향상시킬 필요가 있다.

참고문헌

[1] F. Glover, M. Laguna, "*Tabu Search*," Kluwer Academic Publishers, pp. 1-122, 1997.

[2] 강명주, "시뮬레이티드 어닐링 알고리즘을 이용한 클러스터 기반의 멀티캐스트 라우팅 문제 해법," 한국컴퓨터정보학회논문지, 제 9권, 제 3호, 189-194쪽, 2004년 9월.

[3] M. Mitchell, "*An Introduction to Genetic Algorithms*," MIT Press, pp.1-31, 1997.

[4] L.A. Wolsey, "*Integer programming*," Wiley, pp.91-107, 1998.

[5] R.L.Church, C.S. ReVelle, "The Maximal Covering Location Problem," Regional Science, Vol. 32, No. 1, pp.101-118, Dec. 1974.

[6] 황준하, 류광렬, "승무일정계획의 최적화를 위한 이웃해 탐색 기법과 정수계획법의 결합," 한국정보과학회논문지, 제 31권, 제 6호, 829-839쪽, 2004년 6월.

[7] E. Tsang, "*Foundations of Constraint Satisfaction*," Academic Press Limited, pp.17-27, 1996.

[8] U. Junker, S.E. Karisch, N. Kohl, B. Vaaben, T. Fahle, M. Sellmann. "A framework for constraint programming based column generation," Proc. of CP'99, LNCS 1713, pp.261-274, Oct. 1999.

[9] T. Achterberg, T. Berthold, T. Koch, K. Wolter, "Constraint Integer Programming: A New Approach to Integrate CP and MIP," Proc. of CPAIOR'08, LNCS 5015, pp.6-20, May 2008.

[10] B.T. Downs, J.D. Camm, "An Exact Algorithm for the Maximal Covering Problem," Naval Research Logistics, Vol. 43, No. 3, pp.435-461, Dec. 1996.

[11] 박태진, 황준하, 류광렬, "대규모 Maximal Covering 문제 해결을 위한 유전 알고리즘," 한국정보과학회논문지, 제 31권, 제 5호, 570-576쪽, 2004년 5월.

[12] 황준하, "이웃해 탐색 기법을 이용한 Maximal Covering 문제의 해결," 한국컴퓨터정보학회논문지, 제 11권, 제 1호, 129-138쪽, 2006년 3월.

[13] 황준하, 박춘희, 이용환, 류광렬, "정수계획법과 휴리스틱 탐색기법의 결합에 의한 승무일정계획의 최적화," 정보과학회논문지, 제 8권, 제 2호, 195-205쪽, 2002년 4월.

[14] R. J. Vanderbei, "*Linear Programming: Foundations and Extentions*," Kluwer Academic Publishers, pp.13-24, 1997.

[15] ILOG CPLEX 10.2 Documentation, ILOG, 2007.

저 자 소 개

황 준 하

2002년 : 부산대학교 컴퓨터공학과 공학박사

2002년~현재 : 금오공과대학교 컴퓨터공학부 교수

관심분야 : 인공지능, 최적화, 기계학습, 프로그래밍언어