□ 研 究 論 文 □

韓 國 軍 事 運 營 分 析 學 會 誌
第 25 卷, 第 1 號, 1999. 6. 30

0-1 정수계획법을 위한 사전처리의 구현*
(An Implementation of Preprocessing for 0-1 Integer Programming)

엄순근, 성명기, 박찬규, 박순달**

Abstract

Preprocessing for the 0-1 integer programming can reduce the size of problem instance as well as tighten its linear programming relaxations. In this research, the preprocessing techniques are classified into two categories. First, for the reduction of problem size, there are variable fixing and constraint elimination techniques. Second, for the reduction of feasible region, there are coefficient reduction and Euclidean reduction techniques. These methods are implemented and the effects are shown by experimental results.

* 본 연구는 정보통신부 1997년도 대학기초연구지원사업(97-G0683, C1-98-5183)의 지원을 받았음
* 서울대학교 산업공학과

1. 서 론

현대사회로 오면서 정수계획법으로 모형화되고 있는 문제들이 급격히 늘어나고 있다[1]. 또한 정수계획법은 많은 응용분야를 가지고 있어 그 활용도가 대단히 높다. 따라서 많은 연구자들이 정수계획법을 위한 해법들을 개발하고 효율적인 구현 방법들을 연구하였다[1][8]. 정수계획법 문제의 해법으로는 해를 분지하면서 상한과 하한을 사용하여 해의 타당성을 빨리 판별하고 가능한 가능해를 적게 열거하는 방법인 분지한계법(branch and bound method)과 절단제약식(cut constraint)을 추가함으로써 가능해를 적게 열거하는 방법인 분지절단법(branch and cut method) 그리고 가능해 집합을 절단해 가면서 정수해를 찾는 방법인 절단평면법(cutting plane method) 등이 있다[8].

이들 방법을 구현한 프로그램에서 문제를 풀려고 하면 우선 문제를 입력받아야 하는데, 대형 문제의 경우에는 입력할 자료가 많고, 또 행렬 A의 대부분의 요소가 0이기 때문에 특정한 형태로 문제를 입력하게 된다. 그 중에서 IBM에서 표준으로 정한 MPS 형태가 널리 사용되고 있다[2]. 그런데 문제가 크기 때문에 중복적이거나 비가능인 제약식이 생길 수 있고, 또 변수의 상한과 하한에 오류가 생길 수 있다. 또 행렬 A의 비영요소만을 입력하다 보면 공백행(empty row) 또는 공백열(empty column)이 생길 수 있다[7][9]. 그래서 문제를 입력받은 후, 해법을 수행하기 전에 필요 없는 제약식이나 변수를 제거해야 할 필요가 있는데, 이러한 일을 사전처리(preprocessing)라 한다[2][3][11]. 또한 문제의 비가능성(infeasibility)이나 입력된 문제의 정확성 등을 검사하는 과정도 사전처리에 포함하기도 한다[2].

0-1 정수계획법을 위한 사전처리의 역할은 다음과 같다.

첫째, 주어진 문제를 검사하여 문제 속에 내재되어 있는 비가능성 등 문제의 정확성을 미리 파악하는 것이다[9]. 0-1 정수계획법에서는 문제의 비가능성을 알아내기 위해서 여러 개의 부분 문제를 푸는 과정을 수행하여야 한다. 따라서 문제를 풀기 전에 미리 그 비가능성을 파악할 수 있다면 그 효과는 크다고 할 수 있다. 또한 사전처리를 통한 비가능성 판별은 문제의 어떤 부분에서 비가능이 발생했는지 알 수 있는 이점도 가지고 있다.

둘째, 문제 축소이다. 즉, 중복제약식의 제거 및 변수 값의 고정 등을 통하여 사전처리 된 원래 문제와 동일한 의미를 지니지만 문제의 크기가 줄어든 문제를 만들 수 있다는 것이다[5][6].

셋째, 가능영역(feasible region)의 축소이다. 즉, 계수 축소(coefficient reduction) 등에 의해서 정수해 집합은 같지만 원래 문제의 가능영역을 축소하여 원래 문제와 LP 이완(relaxation) 문제의 차이(gap)를 줄여주어 보다 빨리 최적해에 도달할 수 있다는 것이다[4][7][10].

이와 같이 사전처리로 얻을 수 있는 이익은 입력된 문제의 정확성 검사, 문제 축소, 가능영역의 축소 등을 통해 해법의 수행속도와 안정성을 높이는 효과 등을 들 수 있다. 더욱이 문제를 여러 번 반복해서 풀어야 할 경우 사전처리에 의해서 줄여진 문제를 반복해서 풀게 되면 수행시간을 줄이는 측면에서 더욱 큰 효과를 얻게 된다.

정수계획법에서도 선형계획법을 위한 사전처리 기법들의 개념을 이용하여 대부분 처리를 할 수 있다.

그리고 또한 0-1 정수계획법의 특성 때문에 추가될 수 있는 기법들이 있다. 본 연구에서는 선형계획법을 위한 사전처리 기법들의 개념을 이용한 처리와 함께 주로 0-1 정수계획법의 특성을 이용한 사전처리 기법들을 다룬다.

2. 문제 축소 및 비가능 판정

일반적인 0-1 정수계획법 문제는 (P)와 같다.

$$\text{(P)} \quad \begin{array}{ll} min & \boldsymbol{c}^T\boldsymbol{x} \\ s.t. & A\boldsymbol{x} = \boldsymbol{b} \\ & x_j = 0 \text{ 또는 } 1,\ j=1,\cdots,n \end{array}$$

여기서 A는 $m\times n$ 행렬이고, $\boldsymbol{x}$와 $\boldsymbol{c}$는 n차 벡터, $\boldsymbol{b}$는 m차 벡터이다.

0-1 정수계획법을 위한 사전처리 방법 중 문제 축소 방법은 중복제약식 및 변수 제거 방법 등을 통해서 원래 문제의 크기를 줄이는 방법이다. 이렇게 함으로써 풀게 되는 LP 이완 문제의 개수와 문제를 푸는 시간을 줄이고자 하는 것이 목적이다. 여기에는 제약식의 최대·최소값을 이용한 방법과 목적함수 계수의 부호를 이용한 방법이 있다.

각 방법은 다음과 같다.

2.1 제약식의 최대·최소값을 이용한 방법

문제축소 방법 중의 하나인 제약식의 최대·최소값을 이용한 방법으로 명백한 중복제약식과 변수의 제거 방법 등이 있다. 그리고 이러한 방법을 적용하는 과정에서 주어진 문제의 비가능성을 판단할 수 있다. 변수가 고정되면 목적함수의 상수 값이 변하게 된다. 따라서 사전처리된 문제는 원래 문제와 목적함수 값이 다를 수 있다. 따라서 해법이 종료된 후 해는 복원해 주어야 한다.

2.1.1 중복제약식의 제거

각 변수의 상한과 하한 값을 이용하여 다음과 같이 i번째 제약식이 가질 수 있는 최대값과 최소값을 구할 수 있다.

$$\underline{b}_i = \sum_{j\in N_i} a_{ij}, \quad \overline{b}_i = \sum_{j\in P_i} a_{ij} \qquad (1)$$
단, $P_i = \{j \mid a_{ij} > 0\}$, $N_i = \{j \mid a_{ij} < 0\}$

그러면 다음의 관계가 성립한다.

$$\underline{b}_i \le \sum_j a_{ij}x_j \le \overline{b}_i \qquad (2)$$

i번째 제약식이 '$\le$' 형태인 경우에 우변상수 값에 따라 다음의 결과가 성립한다.

- $b_i < \underline{b}_i$: 비가능.
- $b_i = \underline{b}_i$: 각 변수의 값을 $a_{ij} < 0$이면 1로 $a_{ij} > 0$이면 0으로 고정하고 제약식을 제거한다.
- $\overline{b}_i \le b_i$: 중복제약식이므로 제거한다.
- $\underline{b}_i < b_i < \overline{b}_i$: 현단계에서 제거불능임.

'$\ge$' 형태의 제약식에 대해서는 이와 비슷한 관계가 성립하며, '$=$' 형태의 제약식에 대해서는 '$\le$, $\ge$' 형태의 제약식에 대한 결과가 모두 성립한다.

2.1.2 열의 제거

i번째 제약식을 식 (3)과 같이 표현할 수 있다.

$$\sum_{j\in P_i} a_{ij}x_{ij} + \sum_{j\in N_i} a_{ij}x_{ij} \le b_i \qquad (3)$$

여기서 $j \in P_i$일 때 식 (4)가 성립하면 식 (3)의 모든 0-1 해에서 $x_j = 0$이다. 따라서 변수 x_j의 값을 0으로 고정하고 그 열을 제거한다.

$$a_{ij} > b_i - \sum_{j \in N_i} a_{ij} \quad (4)$$

그리고 $j \in N_i$일 때 식 (5)가 성립하면 식 (3)의 모든 0-1 해에서 $x_j = 1$이다. 따라서 변수 x_j의 값을 1로 고정하고 그 열을 제거한다.

$$-a_{ij} > b_i - \sum_{j \in N_i} a_{ij} \quad (5)$$

'≥' 형태의 제약식에 대해서는 이와 비슷한 관계가 성립하며, '=' 형태의 제약식에 대해서는 '≤, ≥' 형태의 제약식에 대한 결과가 모두 성립한다.

2.1.3 공백행에 대한 처리

공백행에 대해서는 제약식의 형태와 우변상수 값에 따라 그 제약식이 중복인지 비가능인지를 판정한다. 자세한 내용은 다음과 같다.

제약식 형태	우변상수	제약식 상태
≤	< 0	비가능 제약식
	> 0	중복제약식
=	$= 0$	중복제약식
	$\neq 0$	비가능 제약식
≥	> 0	비가능 제약식
	< 0	중복제약식

이 과정들은 초기 단계에서만 행해지는 것만 아니라 다른 기법들의 중간 과정에 나타날 수 있다.

2.2 목적함수 계수의 부호를 이용한 방법

문제축소 방법 중의 하나인 목적함수 계수의 부호를 이용한 방법으로 변수 고정(optimality fixing)과 공백열에 대한 처리가 있다.

2.2.1 변수 고정

제약식이 '≥' 형태인 행 지수들의 집합을 H라고 하고, 제약식이 '≤' 형태인 행 지수들의 집합을 L이라고 하자. 그러면 다음의 두 경우에 변수의 값을 고정할 수 있다.

① If $c_j \geq 0$, $a_i < 0$ for all $i \in H$,
and $a_i > 0$ for all $i \in L$

② If $c_j \leq 0$, $a_i > 0$ for all $i \in H$,
and $a_i < 0$ for all $i \in L$

①의 경우 변수 x_j의 값을 0으로 고정하고, ②의 경우 변수 x_j의 값을 1로 고정한다. 그리고 만약 열 j가 '=' 형태의 행에 비영요소를 가지고 있다면 변수 x_j는 고정할 수 없다.

2.2.2 공백열에 대한 처리

공백열에 대해서는 목적함수 계수의 부호에 따라 해당 변수의 값을 0 또는 1로 고정한다. 변수 x_j의 열이 공백열일 때 다음과 같은 처리하고 변수 x_j를 제거한다.

① $c_j > 0$ 이면 $x_j = 0$으로 고정한다.

② $c_j < 0$ 이면 $x_j = 1$로 고정한다.

③ $c_j = 0$ 이면 임의의 값으로 고정한다.

3. 가능영역의 축소

0-1 정수계획법을 위한 사전처리 방법 중의 하나로 가능영역의 축소 방법이 있다. 이 방법은 정수해

집합에는 영향을 미치지 않으면서 LP 이완 문제의 가능영역을 작게 만들어 줌으로써 원래의 0-1 정수계획법 문제와 LP 이완 문제의 차이를 줄여주게 된다. 이렇게 함으로써 좀더 빨리 최적 정수해에 도달하고자 하는 것이 목적이다. 가능영역의 축소방법으로 계수 축소와 유클리드 축소(Euclidean reduction)가 있다. 각 방법은 다음과 같다.

3.1 계수 축소

식 (6)과 같은 제약식을 생각해보자.

$$\sum_{j\in P_i} a_{ij}x_{ij} \geq a_0 \qquad (6)$$

$k\in P_i$인 어떤 k에 대해서, 만약 $a_{ik} > a_0$이면 a_{ik}를 a_0로 바꾸어 계수 축소를 할 수 있다. 왜냐하면 비영요소가 모두 양수이기 때문에 만약에 x_{ik}가 1의 값을 가진다면 제약식을 만족시키기 위해서 그 계수가 a_0보다 더 큰 값을 가질 필요가 없기 때문이다.

식 (7)과 같이 제약식이 '$\leq$' 형태인 경우에도 이와 유사하게 처리할 수 있다.

$$\sum_{j\in P_i} a_{ij}x_{ij} \leq a_0 \qquad (7)$$

이 경우에도 $k\in N_i$인 어떤 k에 대해서, 만약 $a_{ik} < a_0$이면 a_{ik}를 a_0로 바꾸어 계수 축소를 할 수 있다.

대부분의 제약식이 항상 위와 같은 형태를 갖지는 않는다. 그러나 모두 0-1 변수이기 때문에 필요한 경우에 $x_{ij}' = 1 - x_{ij}$로 치환하여 식 (6) 또는 식 (7)과 같은 형태로 변형할 수 있다. 이에 대해서 자세히 알아보자.

다음과 같은 0-1 정수계획법 문제가 있다고 하자.

$$\begin{aligned} max\ & 4x_1 - 6x_2 + 8x_3 \\ s.t.\ & 4x_1 - 3x_2 + 2x_3 \leq 4 \\ & x_1, x_2, x_3 \in \{0,1\} \end{aligned}$$

이 문제의 LP최적해는 $x_1 = 0.5$, $x_2 = 0$, $x_3 = 1$ 이다.

만약 이 문제에 계수 축소 기법을 적용하면 다음과 같이 된다.

먼저 $x_1' = 1 - x_1$, $x_3' = 1 - x_3$으로 치환하여 식 (7)과 같은 형태로 만든다. 그러면 제약식은 식 (8)과 같이 변형된다. 이렇게 변형하더라도 0-1 정수계획법 문제로서 일반성을 잃지 않는다.

$$\begin{aligned} & -4x_1' - 3x_2 - 2x_3' \leq -2 \qquad (8) \\ & x_1', x_2, x_3' \in \{0,1\} \end{aligned}$$

식 (8)은 식 (7)과 같은 형태이다. 따라서 계수 축소 기법을 적용하면 x_2의 계수 -3을 -2로 축소할 수 있다. 그러면 식 (9)와 같이 된다.

$$\begin{aligned} & -4x_1' - 2x_2 - 2x_3' \leq -2 \qquad (9) \\ & x_1', x_2, x_3' \in \{0,1\} \end{aligned}$$

식 (9)에서 $x_1 = 1 - x_1'$, $x_3 = 1 - x_3'$으로 치환하면 식 (10)과 같이 된다.

$$\begin{aligned} & 4x_1' - 2x_2 + 2x_3' \leq 4 \qquad (10) \\ & x_1, x_2, x_3 \in \{0,1\} \end{aligned}$$

여기서 다시 $x_1' = 1 - x_1$, $x_3' = 1 - x_3$으로 치환하여 식 (7)과 같은 형태로 만든다. 그러면 제약식은 식 (11)과 같이 변형된다.

$$\begin{aligned} & -4x_1' - 2x_2 - 2x_3' \leq -2 \qquad (11) \\ & x_1', x_2, x_3' \in \{0,1\} \end{aligned}$$

식 (11)은 식 (7)과 같은 형태이다. 따라서 계수 축

소 기법을 적용하면 x_1'의 계수 -4를 -2로 축소할 수 있다. 그러면 식 (12)와 같이 된다.

$$-2x_1' - 2x_2 - 2x_3' \leq -2 \quad (12)$$
$$x_1',\ x_2,\ x_3' \in \{0,1\}$$

식 (12)에서 $x_1 = 1 - x_1'$, $x_3 = 1 - x_3'$으로 치환하면 식 (13)과 같이 된다.

$$2x_1 - 2x_2 + 2x_3 \leq 2 \quad (13)$$
$$x_1,\ x_2,\ x_3 \in \{0,1\}$$

이제는 더 이상 계수 축소가 되지 않는다. 따라서 계수 축소를 하고 난 문제는 다음과 같다.

$$\begin{aligned} max\ & 4x_1 - 6x_2 + 8x_3 \\ s.t.\ & 2x_1 - 2x_2 + 2x_3 \leq 2 \\ & x_1,\ x_2,\ x_3 \in \{0,1\} \end{aligned}$$

이렇게 사전처리 된 문제의 LP최적해는 $x_1 = 0$, $x_2 = 0$, $x_3 = 1$ 이다. 즉, 정수해이므로 원래 문제의 최적해가 된다.

3.2 유클리드 축소

유클리드 축소의 절차는 적어도 2개 이상의 1 또는 -1 이외의 비영요소를 갖는 행에 대해서만 실행을 한다. 왜냐하면 이러한 행이 아니면 최대공약수(GCD; Greatest Common Divisor)가 항상 1이 되기 때문에 유클리드 축소 절차를 수행할 필요가 없기 때문이다.

적어도 2개 이상의 1 또는 -1 이외의 비영요소를 갖는 행에서 어떤 유리수의 비영요소를 a_j라고 하자. 그러면 모든 j에 대해서 $a_j * 10^{KEXP}$가 정수인 가장 작은 수를 $KEXP$라고 하자. 유클리드 축소 절차는 각 행의 비영요소들을 한번 검색하면서 그 행에서의 $KEXP$와 GCD를 계산한다. 그리고 나서 제약식의 양변에 10^{KEXP}를 곱하고 나서 GCD로 나누어준다. 이렇게 하더라도 제약식은 원래의 제약식과 동등하다. 즉, 가능영역에 변화가 없다. 그리고 제약식의 형태에 따라 다음과 같이 각 제약식을 처리해 준다.

① 만약 제약식이 '=' 형태이고 우변상수가 정수가 아니면 이 문제는 정수해를 가질 수 없기 때문에 비가능이다.

② 만약 제약식이 '≤' 형태이면 우변상수의 소수점 이하를 버린다.

③ 만약 제약식이 '≥' 형태이면 우변상수의 소수점 이하를 올림 한다.

유클리드 축소 절차에 의해서 제약식을 위와 같이 처리해주면 정수해집합에는 영향을 미치지 않으면서 LP 이완 문제를 더욱 작게 만들어 준다. 즉, 원래의 0-1 정수계획법 문제와 LP 이완 문제의 차이를 줄어준다. 그래서 최적 정수해에 빨리 도달할 수 있게 해준다[7].

식 (14)을 생각해 보자.

$$3.5x_1 + 2.25x_2 \leq 2.31 \quad (14)$$

유클리드 축소 절차에 의해서 $KEXP$는 2이다. 10^{KEXP}를 양변에 곱한다. 그러면 식 (15)를 얻을 수 있다.

$$350x_1 + 225x_2 \leq 231 \quad (15)$$

그리고 식 (15)의 행렬계수의 GCD는 25이다. GCD로 양변을 나누어준다. 그러면 식 (16)을 얻을 수 있다.

$$14x_1 + 9x_2 \leq 9.24 \quad (16)$$

식 (14)와 식 (16)은 동등한 식이다. 즉, 가능영역은

변화가 없다. 하지만 식 (16)에서 행렬계수가 모두 정수이고 변수가 모두 0-1 변수이기 때문에 우변상수를 유클리드 축소를 이용해서 정수화 시켜주면 정수해집합에 영향을 미치지 않으면서 LP 이완 문제를 더욱 작게 만들어 준다. 즉, 다음과 같이 변형하여도 정수해집합에는 영향을 미치지 않는다. 따라서 식 (17)과 같이 쓸 수 있다.

$$14x_1 + 9x_2 \leq 9 \qquad (17)$$

또 다른 예를 보자. MIPLIB 문제 중의 하나인 diamond 문제는 다음과 같다.

$$\begin{aligned} min \quad & -x_1 \\ s.t. \quad & x_1 + x_2 \geq 0.5 \\ & x_1 + x_2 \leq 1.5 \\ & x_1 - x_2 \leq 0.5 \\ & -x_1 + x_2 \leq 0.5 \end{aligned}$$

$x_1,\ x_2$는 0 또는 1

행렬계수가 모두 정수이고 변수가 모두 0-1 변수이기 때문에 우변상수를 유클리드 축소를 이용해서 정수화 시켜주면 정수해집합에 영향을 미치지 않으면서 LP 이완 문제를 더욱 작게 만들어 준다. 즉, 다음과 같이 변형하여도 정수해집합에는 영향을 미치지 않는다.

$$\begin{aligned} min \quad & -x_1 \\ s.t. \quad & x_1 + x_2 \geq 1 \\ & x_1 + x_2 \leq 1 \\ & x_1 - x_2 \leq 0 \\ & -x_1 + x_2 \leq 0 \end{aligned}$$

$x_1,\ x_2$는 0 또는 1

이와 같은 방법으로 diamond 문제에 유클리드 축소 기법을 사용하여 문제를 변형하여 해법을 수행하면, 사전처리를 하지 않았을 때는 LP 이완 문제를 3번 풀어야만 정수최적해가 없다는 것을 알 수 있지만 사전처리에 의해서 축소된 문제를 풀게되면 LP 이완 문제를 1번만 풀면 정수최적해가 없다는 것을 알 수 있다.

4. 실험결과

MIPLIB 문제 중에서 변수 200개 정도의 소형문제에 대해서 CPLEX에서 사전처리의 효과가 있는 [표 1]의 문제를 대상으로 사전처리에 의한 문제 축소와 가능영역 축소 효과를 실험하였고 사전처리를 하지 않았을 때와 사전처리를 하였을 때의 수행시간도 비교하였다. 본 연구에서의 사전처리 방법은 BNC에 구현하여 -O3 옵션을 사용하여 gcc v2.7을 사용해서 컴파일 하여 Sun Ultra 170에서 실험하였다.

[표 1]은 실험문제의 정보이고, [표 2]에서 보는 바와 같이 사전처리를 함으로써 문제를 축소하고 또한 가능영역을 축소하여 원래 문제와 LP 이완 문제의 차이를 줄여 최적 정수해에 빨리 도달할 수 있게 하여 수행시간을 줄일 수 있음을 실험적으로 보였다.

[표 1] 실험문제의 정보

문제 이름	행의 개수	열의 개수	사전처리 결과			
			(1)	(2)	(3)	(4)
diamond	4	2	0	0	4	0
lseu	28	89	0	1	11	25
p0033	16	33	1	1	11	12
p0040	23	40	0	0	0	10
p0201	133	201	0	0	0	72

(1) 제거된 행의 개수
(2) 제거된 열의 개수
(3) 유클리드 축소의 회수
(4) 계수 축소의 회수

[표 2] 사전처리 실험결과

문제 이름	사전처리 전		사전처리 후	
	(1)	(2)	(1)	(2)
diamond	3	0.06	1	0.03
lseu	148870	2143.10	39067	604.54
p0033	3017	17.09	417	2.64
p0040	57	0.48	32	0.28
p0201	1265	219.86	1047	204.39

(1) 풀 LP 이완 문제의 개수
(2) 수행 시간(단위: 초)

5. 결 론

본 연구에서는 0-1 정수계획법에 적용될 수 있는 사전처리 방법들을 문제 축소 방법과 가능영역 축소 방법으로 분류하였다.

문제 축소를 위한 방법으로 제약식의 최대·최소 값을 이용한 중복제약식 및 변수의 제거, 목적함수 계수의 부호를 이용한 변수 고정, 공백열 제거 등을 구현하였고, 또 가능영역을 축소하는 방법으로 계수 축소와 유클리드 축소 방법을 구현하여 실험을 하였다. 사전처리에 의해서 문제를 축소 할 수 있었고 원래 문제와 정수해집합은 같으면서 가능영역을 줄일 수 있었기 때문에 사전처리를 하지 않았을 때와 비교하여 풀어야 할 부문제의 수를 상당히 줄일 수 있었고 수행시간 면에서도 효과가 있다는 것을 실험을 통해 검증하였다.

위 방법들에 대한 실험결과를 통해 0-1 정수계획법 문제를 효과적으로 풀기 위해 사전처리가 필수적이라는 것을 알 수 있었다.

참고문헌

[1] 박순달, 경영과학, 제3판, 민영사, 1998.

[2] 성명기, 박순달, "대형선형계획법문제의 사전처리", 한국경영과학회 '96추계학술대회, 1996, 285-288

[3] 안재근, 김우제, 박순달, "선형계획법 프로그램 개발에 있어서 사전처리에 관한 연구 및 실험결과", 한국경영과학회 '93 추계학술대회, 1993.

[4] Bradley G. H., P. L. Hammer, L. A. Wolsey, "Coefficient Reduction for Inequalities in 0-1 Variables", Mathematical Programming, 1977, 265-282.

[5] Crowder H., E. L. Johnson, M. W. Padberg, "Solving Large Scale Zero-One Linear Programming Problems", Operations Research 3, 1983, 803-834.

[6] Guignard M., K., Spielberg, "Logical reduction method in zero-one programing", Operations Research 29, 1981, 49-74.

[7] Hoffman K. L., M. Padberg, "Improving LP-Representation of zero-one linear programs for branch-and-cut", ORSA Journal on Computing 3, 1991, 121-134.

[8] Johnson E. L., G. L. Nemhauser, M. W. P. Savelsbergh, "Progress in integer programming an exposition", Technical Report, 1997.

[9] Johnson E. L., M. M. Kostreva, U. Suhl, "Solving 0-1 integer programming problems arising from large-scale planning models", Operatinos Research 33, 1997, 803-819.

[10] Martin R. K., L. Schrage, "Subset Coefficient Reduction Cuts for 0-1 Mixed Integer Programming", Operations Research 33, 1985, 505-526.

[11] Savelsbergh, M. W. P., "Preprocessing and Probing Techniques for Mixed Integer Programming Problems", ORSA Journal on Computing 6, 1994, 445-454.

[98년 12월 21일 접수, 99년 4월 26일 최종수정]