International Journal of Pure and Applied Mathematics
Volume 54 No. 3 2009, 345-357

METHODS AND ALGORITHMS FOR SOLVING THE RESOURCE ALLOCATION PROBLEM

Won-Joo Hwang[1] §, R. Enkhbat[2], A. Bayarbaatar[3]
[1]School of Electronics and Telecommunication Engineering
Inje University
607, Obang-Dong, Gimhae, Gyungnam, 621-749, KOREA
e-mail: ichwang@inje.ac.kr
[2,3]School of Mathematics and Computer Science
National University of Mongolia
Ulaanbaatar, MONGOLIA

Abstract: We are concerned with the resource allocation problem which has many applications and is known as NP-hard problem. We show that the resource allocation problem can be solved efficiently by the methods such as the incremental algorithm, and branch and bound depending on the structure of the problem. Special attention will be paid to the resource allocation problem with a separable objective function which reduces to the discrete one. Computational experiments are provided.

AMS Subject Classification: 46N10, 65K05
Key Words: resource allocation problem, Lipschitz continuity, incremental algorithm, dynamic programming, branch and bound method

1. Introduction

In this paper, we address the resource allocation problem with a separable convex objective function. In general, the resource allocation problem is NP-hard.

Received: June 5, 2009

§Correspondence author

We consider three kind of nonlinear programming problems of the following type:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} f_j(x_j), \\ \text{subject to} \quad & \sum_{j=1}^{n} x_j = m. \end{aligned}$$

We show that the resource allocation problem can be solved efficiently by different methods and algorithms depending on the structure of the problem. There is a wide variety of applications. Due to its importance in applications and its simple mathematical structure, many researchers in widely spread areas have studied this problem for more than thirty years.

This paper is organized as follows. In Section 2, we solve the integer programming problem with separable convex objective function using the Incremental algorithm in [4]. This problem is a discrete version of resource allocation problem which plays an important role in economics and management. Ibaraki presents some of important ones. As a result of the additional assumption of convexity, this algorithm can be more efficient than the dynamic programming technique.

In Section 3, we show that the resource allocation problem with separable Lipschitz continuous convex functions can be reduced to the integer programming problem.

In Section 4, we consider the mixed integer resource allocation problem and solve the problem by the branch and bound algorithm. Methods for solving mixed integer nonlinear programming problems are surveyed in [6].

2. Discrete Resource Allocation Problem

2.1. The Incremental Algorithm

We consider the discrete resource allocation problem with a separable convex objective function of the form

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} f_j(x_j), \\ \text{subject to} \quad & \sum_{j=1}^{n} x_j = m, \\ & x_j \in \{0, 1, 2, ..., m\}, \quad j = 1, 2, ..., n, \end{aligned} \tag{1}$$

where m is a given positive integer and $f_j(x_j)$ is convex in x_j. Ibaraki presents an algorithm for problem (1) in [4]. Although this algorithm is very efficient for small N, its running time is not polynomial, the total running time of INCREMENT Algorithm is $O(n + N \log n)$. Define for $j = 1, 2, ..., n$ and $y = 1, 2, ..., m$

$$d_j(y) = f_j(y) - f_j(y-1). \tag{2}$$

By the convexity of f_j, we have

$$d_j(1) \leq d_j(2) \leq ... \leq d_j(m). \tag{3}$$

Theorem 1. (see [4]) *Consider the resource allocation problem of (1) with a separable convex objective function. Let D_m denote the set of m smallest elements in*

$$D = \{d_j(y) | j = 1, 2, ..., n,\ y = 1, 2, ..., m\}, \tag{4}$$

where the elements $d_j(y)$ and $d_{j'}(y')$ with $j \neq j'$ or $y \neq y'$ are consider different even if they have the same value. If $d_j(y) = d_j(y-1)$, $d_j(y-1)$ has higher priority of being chosen in D_m, but otherwise ties are broken arbitrary. Then the x^* defined below is an optimal solution of (1)

$$x_j^* = \begin{cases} 0 & \text{if } d_j(1) \notin D_m, \\ m & \text{if } d_j(m) \in D_m, \\ y & \text{if } d_j(y) \in D_m \text{ and } d_j(y+1) \in D_m. \end{cases} \tag{5}$$

2.2. Incremental Algorithm

Theorem 1 indicates that problem (1) is solved by finding the m smallest $d_j(y)$ in the set D of (4). Since $d_j(y),\ y = 1, 2, ..., m$, are already sorted for each j by (3), we can achieve this task in the "greedy" fashion. This is also called "marginal allocation" or "incremental" algorithm. That is, starting with an initial solution $x = (0, 0, ..., 0)$, one unit of resource is assigned at each iteration to the most favorable activity unit $\sum x_j = m$ is attained.

Procedure INCREMENT

Input: Problem (1) with a separable objective function.

Output: An optimal solution of problem (1)

Step0: Let $x := (0, 0, ..., 0)$ and $k := 1$.

Step1: Find j^* such that $d_{j^*}(x_{j^*} + 1) = \min\limits_{1 \leq j \leq n} d_j(x_j + 1)$ and let $x_j^* := x_j^* + 1$.

Step2: If $k = m$, then halt. The current x is optimal solution. Otherwise, let $k := k + 1$ and return to step 1.

Theorem 2. (see [4]) *Algorithm INCREMENT correctly computes an optimal solution of problem (1) in $O(n + m \log n)$ time.*

3. Lipschitz Optimization

3.1. Problem Formulation

Definition 1. A function f is said to be Lipschitz continuous on an interval R with respect to the variable x if there exists a $L > 0$ such that

$$\|f(x_1) - f(x_2)\| \leq L\|x_1 - x_2\|, \quad \forall x_1, x_2 \in R.$$

Consider the resource allocation problem with a Lipschitz continuous objective function of the form

$$\begin{array}{ll} \text{minimize} & \sum\limits_{j=1}^{n} \varphi_j(y_j) \\ \text{subject to} & \sum\limits_{j=1}^{n} y_j = a, \\ & 0 \leq y_j \leq a, \quad j = 1, 2, ..., n \end{array} \tag{6}$$

where a is a given positive integer, $\varphi_j(y_j)$ are Lipschitz continuous convex functions in y_j. Construct a subdivision as

$$0, \frac{a}{m}, 2\frac{a}{m}, ..., m\frac{a}{m}.$$

Introduce the function $f_i(x_i)$ by

$$f_i(x_i) = \varphi_i(x_i\frac{a}{m}), \quad x_i \in \{0, 1, ..., m\} \ i = 1, ..., n. \tag{7}$$

Let us replace problem (6) with problem (1).

Lemma 1. *A solution to problem (6) can be solved by the INCREMENTAL Algorithm with accuracy $\varepsilon = Ln\frac{a}{m}$.*

Proof. First, we divide the interval $[0, a]$ into m sub-intervals of equal size, that is uniform mesh.

Let $x_i^*, \ i = \overline{1, n}$ and $y_i^*, \ i = \overline{1, n}$ be the solutions of problems (1) and (6), respectively. Choose $x_i, \ i = \overline{1, m}$ such that

$$\sum_{i=1}^{n} x_i = m, \quad x_i \in \{0, 1, ..., m\}, \quad |y_i^* - x_i\frac{a}{m}| \leq \frac{a}{m}.$$

Then

$$0 \le \sum_{i=1}^{n} \varphi_i(x_i^* \frac{a}{m}) - \sum_{i=1}^{n} \varphi_i(y_i^*) = \sum_{i=1}^{n} f_i(x_i^*) - \sum_{i=1}^{n} \varphi_i(y_i^*)$$
$$\le \sum_{i=1}^{n} f_i(x_i) - \sum_{i=1}^{n} \varphi_i(y_i^*) = \sum_{i=1}^{n} \varphi_i(x_i \frac{a}{m}) - \sum_{i=1}^{n} \varphi_i(y_i^*)$$
$$\le \sum_{i=1}^{n} |\varphi_i(x_i^* \frac{a}{m}) - \varphi_i(y_i^*)| \le L \sum_{i=1}^{n} |x_i \frac{a}{m} - y_i^*| \le Ln \frac{a}{m}.$$

It means that if we solve problem (1) with the objective function (7) for $m \ge Ln\frac{a}{\varepsilon}$, then a solution to problem (6) can be found with ε accuracy.

3.2. Dynamic Programming Approach

Problem (6) can be also solved by dynamic programming method in [1]. The use of stages and states to decompose a dynamic programming problem is implemented computationally by means of the so-called recursive equation. The recursive equation allows one to optimize each stage separately. It also keeps track of the cumulative optimal return for all previously considered stages so that when the last stage is reached the total optimal return for the entire problem is available. The representation is recursive because, as will be detailed below, it computes the optimal return for the first i stages from the optimal return for the first $(i-1)$ stages and the return for stage.

Let $f_i(x_i)$ is cumulative optimal return for stages $1, 2, ...,$ and i given the state of the system is x_i, where x_i the capital allocated to stages $1, 2, ..., i$. As discussed above, it is evident that dynamic programming calculation take on a special pattern. That is, for an N-stage problem, the order of computation is

$$F_1 \to F_2 \to ... \to F_{i-1} \to F_i \to ... \to F_N.$$

As an illustration, the recursive equations for on N-stage resource allocation problem are given by

$$\begin{aligned} &F_1(x) = f_1(x), \quad 0 \le x \le x_0, \\ &F_2(x) = \max_{0 \le x_2 \le x_0} (f_2(x_2) + F_1(x - x_2)), \\ &\vdots \\ &F_k(x) = \max_{0 \le x_k \le x_0} (f_k(x_k) + F_{k-1}(x - x_k)). \end{aligned} \tag{8}$$

Example 1. The problem (6) has been solved by dynamic programming

with functions:

$$\begin{aligned}
&f_1(x_1) = \max\{x_1^2 - 6.5, \frac{1}{2}x_1, x_1^2 - 6x_1\},\\
&f_2(x_2) = (x_2 - 4,5)^2,\\
&f_3(x_3) = \max\{-\frac{1}{2}x_3 - 4, \frac{1}{3}x_3 - 6.4, x_3 - 11.7\},\\
&f_4(x_4) = (2x_4 - 3)^2,\\
&f_5(x_5) = \max\{2|x_5| - 9.2, \frac{1}{6}x_5\},\\
&f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3) + f_4(x_4) + f_5(x_5) \longrightarrow \min,\\
&x_1 + x_2 + x_3 + x_4 + x_5 = 31,\\
&0 \le x_j \le 25, \quad j = 1,2,3,4,5.
\end{aligned} \tag{9}$$

Solution is $x^* = (1.17899, 5.02201, 18.16698, 1.61202, 5.02209)$, $f(x^*) = 2.52166$.

4. Mixed Integer Resource Allocation Problem

4.1. The Branch and Bound Method

To solve the mixed integer resource allocation problems (MIRAP) is difficult for a large number of variables, in particular for large numbers of discrete variables. The branch and bound technique is able to solve MIRAP for a certain problem size, whereas the other techniques are not able to solve the problem within limited iteration and time. But even the branch and bound technique requires a lot of iterations and time to solve larger problems. Therefore other techniques, other implementations or solvers should be considered to solve large MIRAP. We consider a mixed integer programming problem with separable convex functions of the form

$$\begin{aligned}
&\text{minimize} \quad \sum_{j=1}^{n} f_j(x_j),\\
&\text{subject to} \quad \sum_{j=1}^{n} x_j = m,\\
&\qquad x_j \in \{l, l+1, ..., u\}, \quad j = 1,2,...,k < n,\\
&\qquad\qquad 0 \le l < u \le m,\\
&\qquad 0 \le x_j \le m, \quad j = k+1, ..., n.
\end{aligned} \tag{10}$$

Here k elements of the vector x are integer variables and other $n-k$ elements of the vector x are continuous variables, and u and l are vectors of upper and lower bounds for the integer variables x. Branch and bound algorithms

such as the ones described in [6] work by explicitly enumerating possible values of the integer variables until an optimal integer solution has been found. A branch and bound algorithm begins by solving the continuous relaxation of the original problem. If a integer variable is fractional at optimality, the algorithm constructs $u+1$ new subproblems, in which the variable is fixed at one of $\{l, l+1, ..., u\}$. The algorithm continues solving subproblems and creating more subproblems as needed until all subproblems have been eliminated from consideration.

A subproblem can be eliminated from consideration if it is infeasible, if the solution to the subproblem has a higher objective value than a known integer solution, or if solution to the subproblem is an integer solution. After all cases have been considered, the optimal solution is simply the best integer solution that was discovered while solving the subproblems. We assume that at least one feasible point is known. In the following algorithm, the set of feasible points found up to the present stage of the procedure is denoted by Q, whereas P stands for the current part of the initial relaxation of D where we want to improve the actual lower bound.

4.2. Algorithm of the Branch and Bound Method

Initialization:
Determine a set P satisfying $D \subset P$ and a set Q satisfying $Q \subset D$;
Set $\gamma \leftarrow min\{f(x) : x \in Q\}$;
Choose $v \in Q$ satisfying $f(v) = \gamma$;
Compute a lower bound $\mu(P) \leq min\{f(x) : x \in D\}$;
Define $M = \{P\}$;
Set $\mu \leftarrow \mu(P)$, $stop \leftarrow false$, $k \leftarrow 1$.
while $stop = false$ do
if $\gamma = \mu$ then $stop \leftarrow true$ (v is an optimal solution, and γ the optimal objective function value of the problem) else
Split P into a finite number of subset, $P_1, ..., P_r$, such that

$$\bigcup_{i=1}^{r} P_i = P, \ intP_i \cap intP_j = 0 \ for \ i \neq j.$$

Compute lower bounds $\mu(P_i)$ for f over $D \cap P_i$ satisfying

$$\mu(P_i) \geq \mu \quad (i = 1, ..., r);$$

Update Q by enclosing all feasible points found while computing the lower bounds $\mu(P_i)$.

Update $\gamma = min\{f(x) : x \in Q\},\ v \in Q$ *satisfying* $f(v) = \gamma$;
Set $M \leftarrow (M \backslash P) \cup \{P_i, ..., P_r\}$;
Delete all sets $P \in M$ *satisfying*

$$\mu(P) \geq \gamma \text{ or } P \cap D = \emptyset;$$

Let R *denote the remaining set;*

$$Set\ \mu \leftarrow \begin{cases} \mu \quad if \quad R = \emptyset \\ min\{\mu(P) : P \in R\}, \quad otherwise \end{cases}$$

Choose

$$P \in R \text{ satisfying } \mu(P) = \mu.$$

end if
Set $M \leftarrow R, k \leftarrow k + 1$.
end while.

5. Numerical Examples

5.1. The Examples of Problem (1)

Since functions f_j are convex, problem (1) can be solved by Incremental algorithm. The following test problems have been solved numerically.

Example 2.

$$\begin{aligned} &f(x) = (x_1 - 2.5)^2 + (2x_2 - 3)^2 + \frac{(x_3 - 7)^2}{8} + \frac{(x_4 - 4.7)^2}{3} + \frac{(x_5 - 1.8)^2}{2}, \\ &x_1 + x_2 + x_3 + x_4 + x_5 = 25, \\ &x_i \in \{1, 2, ..., 25\}, \quad i = 1, 2, ..., 5. \end{aligned} \tag{11}$$

Solution is $x^* = (3, 2, 11, 6, 3),\ f(x^*) = 4.533$.

Example 3.

$$\begin{aligned} &f(x) = \sum_{j=1}^{n} (x_j - j)^2 \longrightarrow \min, \\ &\sum_{j=1}^{n} x_j = n^2, \\ &x_j \in \{0, 1, 2, ..., n^2\}, \quad j = 1, 2, ..., n. \end{aligned} \tag{12}$$

n	f	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
3	3.0	2	3	4	-	-	-	-	-	-	-
4	10.0	2	3	5	6	-	-	-	-	-	-
5	20.0	3	4	5	6	7	-	-	-	-	-
6	39.0	3	4	5	7	8	9	-	-	-	-
7	63.0	4	5	6	7	8	9	10	-	-	-
8	100.0	4	5	6	7	9	10	11	12	-	-
9	144.0	5	6	7	8	9	10	11	12	13	-
10	196.0	5	6	7	8	9	11	12	13	14	15

Table 1: Results of the numerical experiments for Example 3

Example 4.

$$\begin{aligned} &f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 \longrightarrow \min, \\ &x_1 + x_2 + x_3 = 25, \\ &x_j \in \{0, 1, 2, ..., 25\}, \quad j = 1, 2, 3. \end{aligned} \tag{13}$$

Solution is $x^* = (9, 10, 6),\ f(x^*) = 171.375$.

Example 5.

$$\begin{aligned} &f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 + \frac{1}{5}x_4^5 \longrightarrow \min, \\ &x_1 + x_2 + x_3 + x_4 = 25, \\ &x_j \in \{0, 1, 2, ..., 25\}, \quad j = 1, 2, 3, 4. \end{aligned} \tag{14}$$

Solution is $x^* = (8, 9, 6, 2),\ f(x^*) = 144.515$.

Example 6.

$$\begin{aligned} &f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 + \frac{1}{5}x_4^5 + \frac{1}{45}x_5^6 \longrightarrow \min, \\ &x_1 + x_2 + x_3 + x_4 + x_5 = 25, \\ &x_j \in \{0, 1, 2, ..., 25\}, \quad j = 1, 2, ..., 5. \end{aligned} \tag{15}$$

Solution is $x^* = (7, 9, 5, 2, 2),\ f(x^*) = 115.210662$.

Example 7.

$$\begin{aligned} &f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 + \frac{1}{5}x_4^5 + \frac{1}{45}x_5^6 + \frac{1}{135}x_6^7 \longrightarrow \min, \\ &x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 25, \\ &x_j \in \{0, 1, 2, ..., 25\}, \quad j = 1, 2, ..., 6. \end{aligned} \tag{16}$$

Solution is $x^* = (6, 8, 5, 2, 2, 2),\ f(x^*) = 90.138$.

n	3	4	5	6	7	8	9	10
x_1	2.25	2.66	3.12	3.60	4.08	4.57	5.06	5.55
x_2	3.00	3.33	3.75	4.20	5.25	5.71	6.18	6.66
x_3	3.75	4.66	5.00	5.40	5.83	6.28	6.75	7.22
x_4	-	5.33	6.25	6.60	7.00	7.42	7.87	8.33
x_5	-	-	6.87	7.80	8.16	8.57	9.00	9.44
x_6	-	-	-	8.40	8.75	9.71	10.12	10.55
x_7	-	-	-	-	9.91	10.28	11.25	11.66
x_8	-	-	-	-	-	11.42	11.81	12.77
x_9	-	-	-	-	-	-	12.93	13.33
x_{10}	-	-	-	-	-	-	-	14.44
f_{min}	3.13	9.11	20.16	37.72	63.20	98.20	144.23	202.78

Table 2: Results of the numerical experiments for Example 9

5.2. The Examples of Problem (6)

Since functions f_j are Lipschitz continuous convex, we use the incremental algorithm to solve problem (6). Before using the Incremental algorithm, the interval $(0, a)$ is divided into m sub-intervals of equal size, that is uniform mesh.

Example 8.

$$\begin{aligned} &f(x) = (x_1 - 2.5)^2 + (2x_2 - 3)^2 + \frac{(x_3 - 7)^2}{8} + \frac{(x_4 - 4.7)^2}{3} + \frac{(x_5 - 1.8)^2}{2}, \\ &x_1 + x_2 + x_3 + x_4 + x_5 = 25, \\ &0 \le x_i \le 25, \quad i = 1, 2, ..., 5. \end{aligned} \tag{17}$$

If $m = 500$, then a solution is $x^* = (3.05, 1.65, 11.20, 6.25, 2.85)$, $f(x^*) = 3.949583$.

Example 9.

$$\begin{aligned} &f(x) = \sum_{j=1}^{n} (x_j - j)^2 \longrightarrow \min, \\ &\sum_{j=1}^{n} x_j = n^2, \\ &x_j \in \{0, 1, 2, ..., n^2\}, \quad j = 1, 2, ..., n. \end{aligned} \tag{18}$$

Example 10.

$$\begin{aligned}
&f_1(x_1) = \max\{x_1^2 - 6.5, \frac{1}{2}x_1, x_1^2 - 6x_1\},\ f_2(x_2) = (x_2 - 4,5)^2,\\
&f_3(x_3) = \max\{-\frac{1}{2}x_3 - 4, \frac{1}{3}x_3 - 6.4, x_3 - 11.7\},\\
&f_4(x_4) = (2x_4 - 3)^2,\\
&f_5(x_5) = \max\{2|x_5| - 9.2, \tfrac{1}{6}x_5\},\\
&f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3) + f_4(x_4) + f_5(x_5) \longrightarrow \min,\\
&x_1 + x_2 + x_3 + x_4 + x_5 = 31,\\
&0 \le x_j \le 25, \quad j = 1,2,3,4,5.
\end{aligned} \tag{19}$$

If $m = 500$, then a solution is $x^* = (1.17800, 5.02200, 18.16600, 1.61200, 5.02200)$, $f(x^*) = 2.521660$

Example 11.

$$\begin{aligned}
&f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 \longrightarrow \min,\\
&x_1 + x_2 + x_3 = 25,\\
&0 \le x_j \le 25, \quad j = 1,2,3.
\end{aligned} \tag{20}$$

If $m = 49$, then a solution is $(9.18367, 10.20408, 5.61224)$, $f(x^*) = 171.340669$.

Example 12.

$$\begin{aligned}
&f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 + \frac{1}{5}x_4^5 \longrightarrow \min,\\
&x_1 + x_2 + x_3 + x_4 = 25,\\
&0 \le x_j \le 25, \quad j = 1,2,...,4.
\end{aligned} \tag{21}$$

If $m = 49$, then a solution is $x^* = (8.16327, 9.18367, 5.61224, 2.04082)$, $f(x^*) = 143.444012$.

Example 13.

$$\begin{aligned}
&f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 + \frac{1}{5}x_4^5 + \frac{1}{45}x_5^6 \longrightarrow \min,\\
&x_1 + x_2 + x_3 + x_4 + x_5 = 25,\\
&0 \le x_j \le 25, \quad j = 1,2,...,5.
\end{aligned} \tag{22}$$

If $m = 49$, then a solution is $x^* = (6.63265, 8.67347, 5.10204, 2.04082, 2.55102)$, $f(x^*) = 112.228017$.

Example 14.

$$\begin{aligned}
&f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 + \frac{1}{5}x_4^5 + \frac{1}{45}x_5^6 + \frac{1}{135}x_6^7 \longrightarrow \min,\\
&x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 25,\\
&0 \le x_j \le 25, \quad j = 1,2,...,6.
\end{aligned} \tag{23}$$

Integer	f	x_1	x_2	x_3	x_4	x_5
Frac	0.00006	2.507	1.500	6.987	4.702	1.804
x_1	0.26892	2.00	1.512	7.299	4.816	1.871
x_2	1.01787	2.464	2.000	6.710	4.593	1.733
x_3	0.00003	2.495	1.501	7.000	4.698	1.804
x_4	0.03802	2.474	1.492	6.783	5.000	1.750
x_5	0.02332	2.482	1.499	6.874	4.644	2.000

Table 3: Results of the numerical experiments for Example 15

If $m = 49$, then a solution is $x^* = (5.10204, 7.65306, 5.10204, 2.04082, 2.55102, 2.55102)$, $f(x^*) = 87.218929$.

5.3. The Examples of Problem (10)

The following problems have been solved by the branch and bound method.

Example 15.

$$\begin{aligned} &f(x) = (x_1 - 2.5)^2 + (2x_2 - 3)^2 + \frac{(x_3 - 7)^2}{8} + \frac{(x_4 - 4.7)^2}{3} + \frac{(x_5 - 1.8)^2}{2}, \\ &x_1 + x_2 + x_3 + x_4 + x_5 = 25, \\ &x_i \in \{l, l+1, ..., u\}, \quad i = 1, 2, ..., k, \\ &0 \le x_i \le 25, \quad i = k+1, ..., 5. \end{aligned} \tag{24}$$

Example 16.

$$\begin{aligned} &f(x) = (x_1 - 2.5)^2 + (2x_2 - 3)^2 + \frac{(x_3 - 7)^2}{8} + \frac{(x_4 - 4.7)^2}{3} + \frac{(x_5 - 1.8)^2}{2}, \\ &x_1 + x_2 + x_3 + x_4 + x_5 = 25, \\ &x_i \in \{l, l+1, ..., u\}, \quad i = 1, 2, ..., k, \\ &0 \le x_i \le 25, \quad i = k+1, ..., 5. \end{aligned} \tag{25}$$

If $k = 2$, $l = 0$ and $u = 8$ then solution is $x^* = (3.000, 1.000, 7.001, 4.702, 1.797)$, $f(x^*) = 1.2500$. If $k = 3$, $l = 0$ and $u = 8$ then solution is $x^* = (3.000, 1.000, 7.000, 4.700, 1.800)$, $f(x^*) = 1.2500$

Example 17.

$$\begin{aligned} &f(x) = x_1^2 + \frac{3}{50}x_2^3 + \frac{3}{128}x_3^4 + \frac{1}{5}x_4^5 \longrightarrow \min, \\ &x_1 + x_2 + x_3 + x_4 + x_5 = 25, \\ &x_i \in \{l, l+1, ..., u\}, \quad i = 1, 2, ..., k, \\ &0 \le x_i \le 25, \quad i = k+1, ..., 5. \end{aligned} \tag{26}$$

A fractional solution of the original problem is $x^* = (3.3806, 6.1285, 4.1695, 1.6249, 2.1965)$, $f(x^*) = 37.0836$ If $k = 1$, $l = 0$ and $u = 8$ then a solution is $x^* = (3.0000, 6.3630, 4.2693, 1.6407, 2.2270)$, $f(x^*) = 37.3325$. If $k = 2$, $l = 0$ and $u = 8$ then a solution is $x^* = (3.000, 6.000, 4.4951, 1.7093, 2.2956)$, $f(x^*) = 37.6994$.

Acknowledgements

This work supported by Korea Science and Engineering Foundation through the joint Research Program(Grant No.F01-2008-000-10196-0).

References

[1] D.P. Bertsekas, *Dynamic Programming and Optimal Control*, Second Edition, Vol-s: I and II, Athena Scientific (2002).

[2] B. Brian, J.E. Mitchell, An improved branch and bound algorithm for mixed integer nonlinear programs, *Computers and Operation Research*, **21** (1994), 359-367.

[3] R. Enkhbat, An algorithm for maximizing a convex function over a simple set, *Journal of Global Optimization*, **8** (1996), 379-391.

[4] T. Ibaraki, N. Katoh, *Resource Allocation Problems, Algorithmic Approaches*, The MIT Press, Cambridge (1988), 53-78.

[5] D.J. Laughhunn, Quadratic binary programming with applications to capital-budgeting problems, *Operations Research*, **18** (1979), 454-461.

[6] K.G. Omprakash, A. Ravindran, Branch and bound experiments in convex nonlinear integer programming, *Management Science*, **31** (1985), 1533-1546.

[7] A.G. Suxarev, A.V. Timoxov, V.V. Fedorov, *Approximation Theory and Optimization*, A book in Russian (2005), 328-331.