韓國 軍事運營分析 學會誌
第 11 券 第 2 號, 1985.12.

多制約式下에서의 最適重復設計에 관한 硏究 **

(Redundancy Optimization under Multiple Constraints)

尹 德 均 *

Abstract

This paper presents a multi-costraint optimization model for redundant system reliability. The optimization model is usually formulated as a nonlinear integer programming (NIP) problem. This paper reformulates the NIP problem into a linear integer programming (LIP) problem. Then an efficient "Branch and Straddle" algorithm is proposed to solve the LIP problem. The efficiency of this algorithm stems from the simultaneous handling of multiple variables, unlike in ordinary branch and bound algorithms. A numerical example is given to illustrate this algorithm.

1. 서 론

다단계로 이루어진 시스템의 설계에 있어서 고신뢰도를 보장하기 위해서는 시스템의 주요 부품의 중복 설계를 실시하여야 한다.

그러나 동시에 이러한 중복 설계에 있어서는 비용, 무게, 부피라는 각 요소에 대한 제약조건이 따르게 된다. 그 때의 문제는 우리가 어떻게 최적 중복 설계를 각 단계에 배분하느냐 하는 것이다.

다시 말하면 비용, 무게, 부피 등을 허용된 범위내에서 시스템의 신뢰성을 최대화하기 위해서 어떻게 중복 설계를 배분할 것이냐 하는 것이다.

이러한 중복설계 문제는 Moskowitz 등 [9]이 처음으로 변분학적 방법을 사용하여 문제를 연구한 이래 많은 연구의 대상이 되어왔다. Tillman 등 [15]은 문헌조사에서 이러한 최적화문제에 대하여 이 해법을 2가지로 대별하고 있다. 짧은 계산 수행시간에 비하여 적절한 근사해를 주는 근사해법 (Aggarwal [1], Nagakawa와 Nagashima [11], Narasimhalu와 Si-

* 漢陽大學校 工科大學
** 本 硏究는 韓國科學財團의 支援에 의한 것임

varamakrishnan [13]) 과 계산 수행시간은 많이 걸리나 정확한 해를 구하는 완전해법 (Bellman 과 Dreyfus [2], Tillman [14], Misra [8], Hyun [4], Luus [6], Mcleavy [7]) 이다.

일반적으로 제약식이 많을 경우 동적계획법보다는 정수계획법이 유리하다. 그러나 정수계획법은 變數가 늘어남에 따라 기하급수적으로 컴퓨터 수행시간이 증가하는 결점이 있다.

동 연구에서 제시한 해법은 근본적으로는 0-1 정수계획법 중 Yun 등 [16] 의 선형배낭기법에 기초를 두고 이를 다제약식으로 확장한 것이다. 정수계획법의 계산상의 결점을 극복하기 위해서는 효율적인 브랜치와스트래들(Branch and Straddle) 알고리즘을 사용하고 있다. 브랜치와스트래들 기법은 근본적으로는 분지한계법과 맥락을 같이 한다.

이 기법의 장점은 분지 과정에서 다변수를 처리하는 데서 분지한계법보다 우수한 기법으로 입증되고 있다 (Yun 등 [16] 1982).

2. 기본 수학 모델

본 논문에서 사용하는 수학 기호는 다음과 같다.

n_i : 서브시스템에서의 중복설계 부품의 수

c_{im} : 서브시스템의 부품당의 m자원의 소요량

b_m : 중복설계부품에 대한 가용 잉여자원

p_i, g_i : 서브시스템 i의 부품의 신뢰도, 비 신뢰도

$R(n)$: n의 함수로서 시스템신뢰도

t_i : n_i의 상한값

시스템이 N단계의 부분시스템으로 이루어져 있고 그 i단계의 부분시스템은 $g_i(=1-p_i)$의 비신뢰도 n_i+1개의 부품으로 구성되어 있다고 하자.

각 단계의 모든 부분품이 고장이 날 경우에 그 단계의 부분시스템은 고장이 나고 어느 한 단계의 부분시스템이 고장이 날 경우에는 전체시스템이 고장이 나는 소위 말하는 병렬-직렬 (Parallel-Series)의 전형적인 시스템이다.

이를 수식화하면 다음과 같은 비선형 정수계획법이 된다.

목적함수는

$$\text{maximize } R(n)=\prod_{i=1}^{N}(1-g_i^{n_i+1})$$

$$=\prod_{i=1}^{N}(1-g_i)[(1-g_i)^2/(1-g_i)]\cdots[(1-g_i^{n_i+1})/(1-g_i^{n_i})] \quad \cdots\cdots(1)$$

선형제약식은

$$\sum_{i=1}^{N} c_{im}n_i \leqq b_m,\ m=1,2,\cdots\cdots M,$$

$$0\leqq n_i\leqq t_i,\ n_i=\text{정수} \quad \cdots\cdots(2)$$

단 t_i는 n_i의 상방한계치

만일 t_i가 주어지지 않을 경우 제약식 (2) 로부터

$$t_i=\min_m [\,b_m/c_{im}\,] \quad \cdots\cdots(3)$$

단 [ ] 는 정수값 표시

3. 비선형정수계획법의 선형정수계획법으로의 전환

X_{ik}를 0,1을 갖는 변수로 정의하자. 이의 의미는 서브시스템 i의 k번째 부품이 설치된다고 하면 X_{ik} 값은 1을, 설치되지 않을 경우 0을 갖게 된다.

그러므로 변수 X_{ik}와 n_i 사이에는 다음과 같은 관계식이 성립된다.

$$\sum_{k=1}^{t_i} X_{ik} = n_i \quad \cdots\cdots (4)$$

$$X_{i,\,k+1} \leqq X_{ik}, \forall_{i,k} \quad \cdots\cdots (5)$$

다시말하면 다음 관계가 된다.

$X_{ik} = 1$ 만일 $k \leqq n_i$

$X_{ik} = 0$ 만일 $k > n_i$

식 (1)의 n을 (5)식의 X로 대치하면

$$R(x) = \pi_{i=1}^{N} \pi_{k=1}^{t_i} p_i[1-g_i^{k+1}]/(1-g_i^{k})]^{x_{ik}} \quad \cdots\cdots (6)$$

만일 양변에 대수변환을 취하면

$$\ln R(x) = \sum_{i=1}^{N} \sum_{k=i}^{t_i} a_{ik}\, x_{ik} + c$$

단, $a_{ik} = \ln[(1-g_i^{k+1})/(1-g_i^{k})]$

$$c = \sum_{i=1}^{N} \ln p_i$$

상수항 c를 제거하고 나면 비선형정수계획법의 식(1)은 다음과 같이 선형정수계획법의 문제로 공식화된다.

$$\text{maximize} : f(x) = \sum_{i=1}^{N} \sum_{k=1}^{t_i} a_{ik}\, x_{ik} \quad \cdots\cdots (9)$$

$$\text{s.t.} : \sum_{i=1}^{N} \sum_{k=1}^{t_i} c_{im}\, X_{ik} \leqq b_m$$

$$m = 1, 2, \cdots\cdots M \quad \cdots\cdots (10)$$

$$X_{i,\,k+1} \leqq X_{ik}, \forall_{i,k} \quad \cdots\cdots (5)$$

$$X_{ik} = 0, 1$$

4. 해법논리

본 문제를 풀기 위해서는 (5)식을 무시하고 기존의 0-1 정수계획법 ([3], [5], [12], [13])을 사용하면 된다. 본 문제의 구성상 정수계획으로 풀면 (5)식은 자동적으로 만족되도록 되어있다. 즉 a_{ik}의 값은 k에 대해서 감소함수가 되며 C_{im}은 k에 대해서 일정하다. 그러므로 정수계획법의 최대화의 논리상 X_{ik}가 1이 안되면 X_{ik+1}이1이 되는 일은절대로 없게 된다. 배낭문제에서 같은 값인데 효용이 큰 것이 포함되지 않은데 작은것이 포할될 수는 절대로 없기 때문이다.

그러나 Yun 등[16]이 단일 제약식문제에서 보인 바와 같이 (5)식의 제약조건을 이용하여 분지한계법 (Branch and Bound) 기법에서는 다변수를 취급할 수 있게 되는 것이다. 그러므로 본 논문에서의 해법은 (5)식의 제약요소를 이용하여 다변수를 취급하는 Branch and Straddle 논리에 기초를 두고 있으나 다제약식에의 적용을 위해서 Shih[13]의 다제약식하에서 해법을 원용하고 있다.

요약하여 말하면 분지한계법에서 노드의 상한값을 구하는데 다제약식의 (9), (10)의 한개의 최대화 식을 푸는 대신 각각의 자원 m에 대하여 목적함수는 동일하

고 제약식이 다른 단일제약식의 최대화 문제를 풀도록 하는 것이다.

즉 각각의 자원 m에 대한 다음 배낭 문제를 풀게 된다.

$$\text{maximize}: f(x) = \sum_{i=1}^{N} \sum_{k=1}^{t_i} a_{ik}\, x_{ik} \quad \cdots\cdots(11)$$

$$s.t.: \sum_{i=1}^{N} \sum_{k=1}^{t_i} C_{im}\, X_{ik} \leqq b_m \quad \cdots\cdots(12)$$

$$X_{i,k+1} \leqq X_{ik}\ , \forall_{i,k} \quad \cdots\cdots(13)$$
$$X_{ik} = 0, 1$$

먼저 식(11)과 (13)으로 구성되는 배낭문제의 해법을 생각해 보자. 서브시스템 i에서 k번째의 부품의 설치는 c_{im} 단위의 자원을 소비하며 a_{ik} 만큼의 시스템의 신뢰도를 증가시킨다. 그러므로 서브시스템 i의 k번째 부품의 설치의 한계효용은 c_{im}에 대한 a_{ik}의 비로서 표현된다. e_{ikm}를 한계효용 (a_{ik}/c_{im})이라 하자. 이 배낭문제의 상한값을 구하기 위하여 먼저 한계효용 (e_{ikm})을 감소하는 순서로 참고표를 만든다. 이 참고표를 역순으로 하여 한계효용 순위수를 정한다. $\sigma_m(\ell)$은 자원 m에 의한 한계효용의 순위가 ℓ인 부품의 서브시스템을 나타낸다고 하자. 같은 방법으로 $\sigma_m(\ell)$은 이 서브시스템에서 한계효용 순위가 ℓ인 부품의 수라 하자. 그렇다면 $\sigma_m(3)$ $\sigma_m(3) = (2, 5)$ 자원 m의 한계효용의 우선순위가 3인 부품이 2번째 서브시스템의 5번째 부품을 나타낸다. 그러므로 식 (11), (13)은 다음과 같이 다시 쓸수 있다.

$$\text{maximize}: f(x) = \sum_{\ell=1}^{t} a_{\sigma_m(\ell)\,\rho_m(\ell)} \cdot X_{\sigma(\ell)\rho(\ell)} \quad \cdots\cdots(14)$$

$$s.t.: \sum_{\ell=1}^{t} C_{\sigma_m(\ell)}\, X_{\sigma_m(\ell)\,\rho_m(\ell)} \leqq b_m \quad \cdots\cdots(15)$$

$$X_{\sigma_m(\ell)\ \rho_m(\ell)+1} \leqq X_{\sigma_m(\ell)\ \rho_m(\ell)}$$

$$\forall \ell \quad \cdots\cdots(16)$$

$$X_{\sigma_m(\ell)\ \rho_m(\ell)} = 0, 1$$

여기서 $T = \sum_{i=1}^{N} t_i$.

본 논문의 Branch and Straddle 논리는 가능한 해의 영역을 점차 작게 세분하여 제거하는 반복하는 과정을 통하여 최적해를 구한다. Branch and Straddle 논리는 해의 서브클래스를 나타내는 노드(node)의 tree를 사용하여 표시한다. 노드 j는 해로부터 제외하기로 결정된 부품과 포함시키기로 결정된 부품의 집합으로 설명되어진다. 즉 주어진 노드 j에 대하여 3가지의 부품의 집합이 있다.

(1) I(j)는 노드 j에서 포함되기로 결정된 부품의 집합

(2) E(j)는 노드 j에서 제외하기로 결정된 부품의 집합

(3) $(I(j) \cup E(j))^c$는 결정되지 않은 부품의 집합

이다.

노드 번호는 특별한 의미가 있는 것이 아니고 발생순서대로 부친다. 분지되지 않는 노드를 터미날노드라 한다.

각각의 터미날노드에 대하여 다음에서 설명하겠지만 목적식에 대한 상한치(U(j))와 하한치(L(j))가 계산되어 진다. 분지는 현재 터미날 노드 중에서 최고

의 상한치 $U^*(=\max\{U(j)\})$를 갖는 터미날 노드에서 두개의 상이한 터미날 노드를 만들게 된다. 각각의 터미날 노드에 대하여 계산한 하한치 중에서 최대 하한치를 계산하여 그것보다 작은 상한치를 갖는 노드는 제거한다. Branch and Straddle 알고리즘은 이와같이 상한과 하한값을 동시에 계산함으로서 2가지의 장점이 있다. 첫째 현재 하한치 해는 중간 가능해이다. 그러므로 이 해가 현재의 상한치와 만족할 만큼 근접해 있다면 중간에서 계산을 끝마칠 수 있다. 그리고 현재의 최대하한치 L^*보다 작은 상한치를 가지는 분지들을 제거함으로서 컴퓨터의 기억소용량을 줄일 수 있다.

각각의 노드 j에 있어서 I(j)와 E(j)의 요소가 미리 결정되기 때문에 (i,e $X_{\sigma_m(\ell)\rho_m(\ell)}=1 \; \forall \; \sigma_m(\rho)\,\rho_m(\ell) \in I(j)$ and $X_{\sigma_m(\ell)\rho_m(\ell)}=0 \; \forall \; \sigma_m(\ell)\,\rho_m(\ell) \in E(j)$) 결정되지 않은 요소 중에서 노드 j의 남은 자원, $b_m(j)$를 활용할 수 있는 요소를 선택하는 것이다.

$$b_m(j)=b_m-\sum c_{im} X_{\sigma_m(\ell)\rho_m(\ell)} \cdots (17)$$
$$\forall\ \sigma_m(\ell)\,\rho_m(\ell) \in I(j)$$

노드의 하한치는 다음과 같이 구해진다. 각각의 자원에 대하여 한계효용의 내림차 순으로 정리하여 아직 미정의 부품 중에서 소진할 때까지 한계효용이 높은 순서로 부품을 선택한다. 남은 자원을 이용하기 위하여 먼저 남은 부품 중에서 최대의 한계효용을 주는 부품을 선택한다. 만일 남은 자원이 그 다음 부품을 허락한다면 다시 두번째 부품을 선택한다. 부품을 선택하게 되면 그만큼 각각의 자원을 감소시킨다. 이 절차는 다음식으로 나타낼 수 있다.

$$X_{\sigma_m(\ell)\rho_m(\ell)}=\begin{cases}1 \text{ if } b_m(j)-\sum_{\ell\leqq r} X_{\sigma_m(\ell)\rho_m(\ell)}\, c_{\sigma_m(\ell)} \\ \quad \geqq 0 \text{ for } \ell\in[I(j)\cup E(j)]^c \\ \quad \text{for all } m \\ 0 \text{ otherwise} \end{cases} \cdots\cdots (18)$$

목적식에 관계된 하한치는

$$L_m(j)=\sum_{\ell=1}^{T} X_{\sigma_m(\ell)\rho_m(\ell)}\, a_{\sigma_m(\ell)\rho_m(\ell)} \quad (\ell) \cdots\cdots (19)$$

이 다음 노드의 하한치는 자원m에 대한 $L_m(j)$ 중에서 최대치가 된다.

$$L(j)=\max_m L_m(j) \cdots\cdots (20)$$

노드 j 에서 상한치를 계산하기 위한것은 다음 문제를 푸는 것으로 충분하다.

$$\text{Maximize}: f(X)=\sum_\ell X_{\sigma_m(\ell)\rho_m(\ell)}\, a_{\sigma_m(\ell)\rho_m(\ell)}$$
$$\text{for all } \ell\in[I(j)\cup E(j)]^c \cdots\cdots (21)$$

$$s.t: \sum_\ell X_{\sigma_m(\ell)\rho_m(\ell)}\, c_{\sigma_m(\ell)} \leqq b_m(j)$$
$$0 \leq X_{\sigma_m(\ell)\rho_m(\ell)} \leq 1$$

Greenberg(1970)와 Kolesar(1967)에 의해 보여진 바와 같이 식(21)의 해는

$$X_{\sigma_m(\ell)\rho_m(\ell)}= 1 \text{ if } \ell<\alpha \text{ for } \ell\in[I(j)\cup E(k)]^c$$
$$0 \text{ if } \ell>\alpha \cdots\cdots (22)$$

$$X_{\sigma_m(\ell)\rho_m(\ell)}=[b_m(j)-\sum_{\ell<\alpha} c_{\sigma_m(\ell)}]/c_{\sigma_m(\alpha)}$$

단 α는 다음식을 만족하는 최소의 수

이다.

$$\sum_{\ell \leqq \alpha} C_{\sigma_m(\ell)}\, \rho_m(\ell) > b_m(j) \quad \text{for}$$

$$\ell \in [I(j) \cup E(j)]^c \quad \cdots\cdots (23)$$

노드 j 에서 목적식에 대한 상한치는

$$U_m(j) = \sum_{\ell=1}^{T} a_{\sigma_m(\ell)\rho_m(\ell)}\, X_{\sigma_n(\ell)} \Big/ \rho_n(\ell) \quad \cdots\cdots (24)$$

이다.

U(j)는 노드 j 에서 상한치 $U_m(j)$ 중에서 최소값을 구하면 된다.

$$U(j) = \min_m U_m(j) \quad \cdots\cdots (25)$$

각각의 과정에서 모든 터미날노드에 대하여 현재의 최대상한치 $U^*(\max_j \{U(j)\})$와 최대하한치 $L^*(\max_j \{L(j)\})$를 결정한다. j^*를 현재의 상한치 U^*가 주어진 터미날노드를 나타낸다.

만일 $U^* = L^*$라 하면 최적해에 이른 것이 된다. 그렇지 않으면 다음 분지는 노드 j^*에서 이루어진다. $\sigma(j^*)$ 노드 j 에서 $X_{\sigma(\ell)\rho(\ell)}$가 소수값을 갖는 부품이라 하자. 그러면 $\sigma(j^*)\rho(j^*)$는 분지의 변수로 선택한다. 여기에서 부분집합 A (j*) (or B(j*)는 서브시스템 $\sigma(j^*)$에서 $\sigma(j^*)\rho(j^*)$ 보다 한계효용이 큰(또는 작은) 부품을 나타낸다. 노드 j0는 A(j*)의 부품은 빼기로 하고 node j1에서 B(j*)의 부품들을 포함시키는 그림 1 의 분지과정을 보여준다.

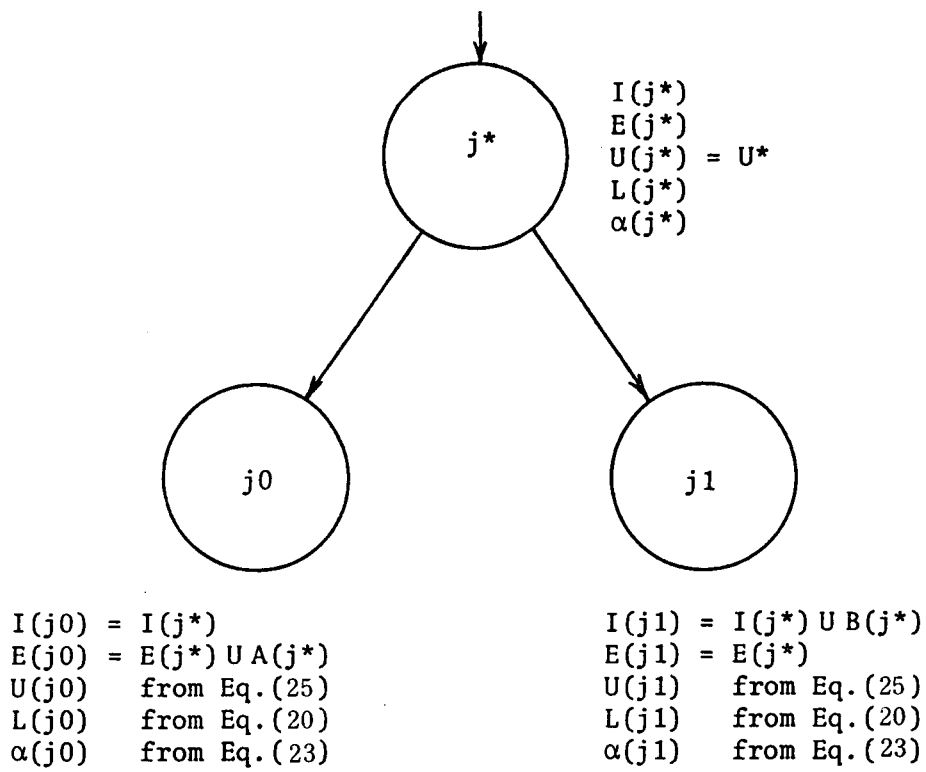

그림 1 . Branching Procedure of a Binary Tree

가. Branch and Straddle 논리 요약

단계 1 : 주어진 데이타 g_i, c_{im}와 b_m에 대하여 t_i, a_{ik} 그리고 e_{ikm}을 결정한다. 자원 m 각각에 대하여 e_{ikm}의 내림차 순으로 참고표를 만든다.

단계 2 : 먼저 노드 번호를 1로 하고 $I(1)=\phi$, $E(1)=\phi$ 세트한다. 상한치 해를 얻기 위하여 식 (21),~(25)를 사용하고 하한치 해를 구하기 위하여 식 (18)~ (20)을 사용한다.

$U^*=U(1)$; $L^*=L(1)$; $j^*=1$이라 하자.

단계 3 : 그렇지않으면 노드 j^*에 대하여 두개의 노드 j0와 j1을 만든다. $\alpha(j^*)$로 부터 $A(j^*)$와 $B(j^*)$를 결정한다.

① $I(j0)=I(j^*)$로 하고 $E(j0)=E(j0)UA(j^*)$로 한다. 식 (21)~ (25)를 사용하여 상한치 해를 구한다. 그리고 식 (18)~ (20)을 사용하여 하한치 해를 구한다.

② $I(j1)=I(j^*)$ $UB(j^*)$ 그리고 $E(j1)=E(j^*)$로 한다. 식 (21)~ (25)를 사용하여 상한치 해를 구한다. 그리고 식 (21)~ (25)를 사용하여 상한치 해를 구한다. 그리고 식 (18)~(20)을 사용하여 하한치 해를 구한다.

단계 4 : 2개의 노드로 분지된 노드 j^*를 터미날 노드에서 제거한다. 그리고나서 터미날 노드에 대하여 현재의 U^*와 L^* j^*를 재결정한다. $U(j)$가 L^* 보다 작은 모든 노드를 제거한다. 단계 3으로 간다.

단계 5 : 최적해 X^*로부터 n^*의 해를 구한다. 해 n^*에 대한 최적신뢰도를 결정한다.

나. 수치예제

선형배낭문제를 설명하기 위하여 표 1에 주어진 3개의 서브시스템 문제를 풀어보자.

<표 1> Date for the Example Problem

Stage i	1	2	3	Available extra resources
Component cost	$4	$5	$6	$20
Component weight	3g	4g	2g	15g
Reliability	.9	.8	.7	

원래의 비선형 정수계획법 문제는 다음과 같다.

$$\text{Maximize } R(n)=(1-.1^{n_1+1})(1-.3^{n_3+1}) \quad \cdots\cdots(26)$$

$$\text{s.t. } 4n_1+5n_2+6n_3 \leqq 20,$$

$$3n_1+4n_2+2n_3 \leqq 15$$

$$n_i = \text{integer}.$$

식 (3)으로 부터 n_i의 상한치를 계산한다. $t_1=5$, $t_2=3$, $t_3=3$

식 (26)의 NIP 문제를 선형 배낭문제를 바꾸기 위하여 식 (8)로 부터 a_{ik}를 계산한다. 이것은 다음표와 같다.

식 (26)의 비선형정수계획법 문제는 다음과 같은 선형배낭문제로 재구성된다.

$$\text{Maximize } f(X)=$$
$$.0953X_{11}+.00905X_{12}+.0009X_{13}+.$$
$$.00009X_{14}+.000009X_{15}+.18232X_{21}+$$
$$.03279X_{22}+.00646X_{23}+.00128X_{24}+$$
$$.26236X_{31}+.6694X_{32}+.01924X_{33}$$

Stage (i)	Number of redundant component (k)				
	1	2	3	4	5
1	.0953	.00905	.00090	.00009	.000009
2	.18232	.03279	.00643		
3	.26232	.06694	.01924		

$$\text{s.t.}: 4(X_{11}+X_{12}+X_{13}+X_{14}+X_{15})+ +5(X_{21}+X_{22}+X_{23}+X_{24})+6(X_{31}+X_{32}+X_{33})\leqq 20 \quad \cdots\cdots(27)$$

$$3(X_{11}+X_{12}+X_{13}+X_{14}+X_{15})+4(X_{21}+X_{22}+X_{23})+2(X_{31}+X_{32}+X_{33})\leqq 15$$

$X_{i,k+1}\leqq X_{ik}$ for all i, k

$X_{ik}=0,1$

Branch and Straddle 알고리즘을 적용하기 위하여 부품들을 한계효용의 내림차순으로 나열하고 다음과 같이 2개의 표에 부품의 순위를 나타냈다.

Look up Table 1

Rank order ℓ	Redundancy(i,k)	c_{ik}	a_{ik}	e_{ikm}
1	(3,1)	6	.26236	.043727
2	(2.1)	5	.18232	.036464
3	(1,1)	4	.09531	.023827
4	(3,2)	6	.06694	.011567
5	(2,2)	5	.03279	.006558
6	(3,3)	6	.01924	.003207
7	(1,2)	4	.00905	.002263
8	(2,3)	5	.00643	.001286
9	(1,3)	4	.00090	.000225
10	(1,4)	4	.00009	.000023
11	(1,5)	4	.00001	.000002

다음과 같은 Branch and Straddle 알고리즘을 사용하여 식 (27)의 선형배낭문제를 푼다.

단계 2 에서 $I(1)=E(1)=\phi$ 이다. 식(18) ~ (25)를 이용하여

비용의 제약식에 대해서 풀면

$U_1(1)=0.596$, $L_1(1)=0.573$, $\alpha_1(1)=(3,2)$

무게의 제약식에 대해서 풀면

$U_2(1)=0.635$, $L_2(1)=0.512$, $\alpha_2(1)=(2,2)$

노드의 상한값과 하한값을 구하면

$U(1)=\min\{0.596, 0.635\}=0.596$

Look up Table 2

Rank Order	Redundancy	c_{i2}	a_{ik}	ρ_{ik2}
1	(3,1)	2	0.26236	0.13118
2	(2,1)	4	0.18232	0.04558
3	(3,2)	1	0.06694	0.03347
4	(1,1)	3	0.09531	0.03177
5	(3,3)	2	0.01924	0.00962
6	(2,2)	4	0.03279	0.0081975
7	(1,2)	3	0.00905	0.0030167
8	(2,3)	4	0.00643	0.0016075
9	(1,3)	3	0.00090	0.00030
10	(1,4)	3	0.00009	0.00003
11	(1,5)	3	0.00001	0.00003

$L(1)=\max\{0.573, 0.512\}=0.573$

$\alpha(1)=(3,2)$로 부터 $A(1)=\{(3,2),(3,3)\}$, $B(1)=\{(3,1),(3,2)\}$

단계 3 에서 $U^* > L^*$ 이므로 분지과정은 계속된다. 처음에 분지는 부품 $\alpha(1) = (3,2)$ 가 사용된다. 노드 $1 = j^*$ 에 대하여 $j0 = 2$, $j1 = 3$을 만든다. $\alpha(1) = (3,2)$ 이므로 $A(j^*) = \{(3,2),(3,3)\}$ $B(j^*) = \{(3,1),(3,2)\}$ 이다.

① 노드 2 에 대하여 $E(2) = \{(3,2),(3,3)\}$ $I(2) = \phi$가 된다. 식 (18)~(25)로 부터 $U_1(2) = 0.573$, $U_2(2) = 0.579$

비용제약식에 대해서 풀면 $U_1(2) = 0.573$, $L_1(2) = 0.573$, $\alpha_1(2) = \phi$

무게제약식에 대해서 풀면 $U_1(2) = 0.579$, $L_2(2) = 0.573$, $\alpha_2(2) = (1,2)$

노드의 상한값과 하한값을 구하면

$U(2) = \min\{0.573, 0.579\} = 0.573$

$L(2) = \max\{0.573, 0.573\} = 0.573$

$A(2) = \{(1,2), (1,3), (1,4)\}$

$B(2) = \{(1,1), (1,2)\}$

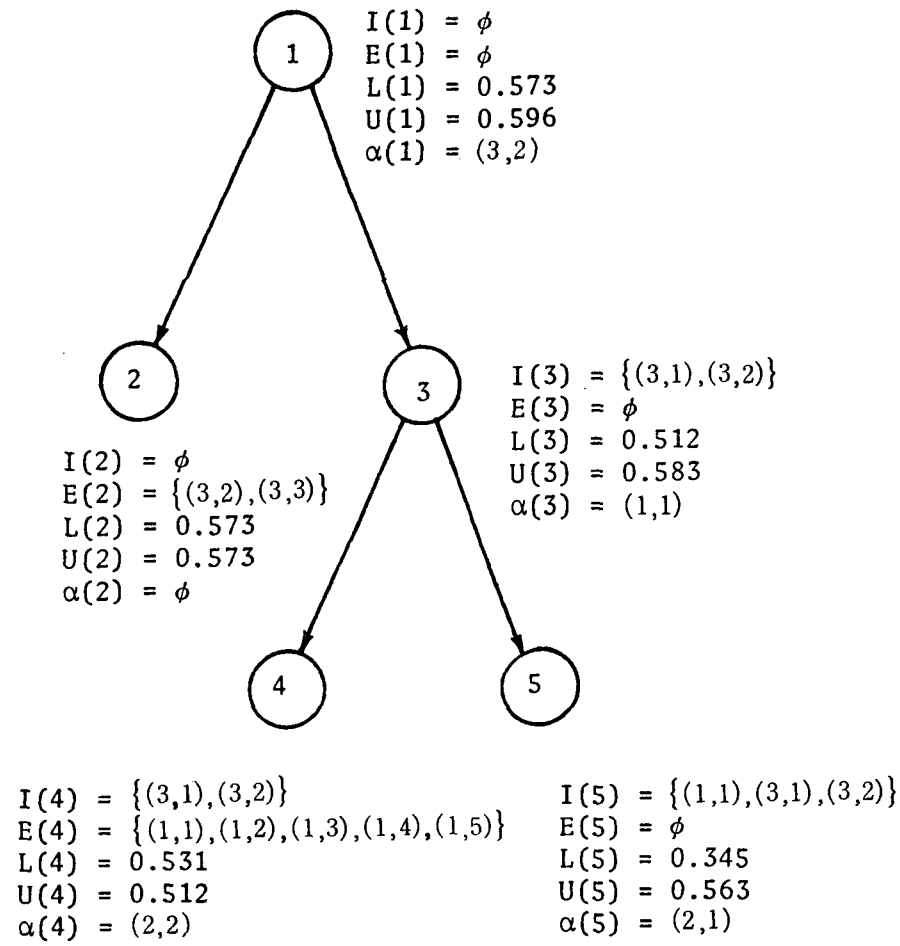

그림 2. Complete Tree in the Branch and Straddle Algorithm for the Example Problem

② 노드 3 에 대하여 $E(3)=\phi$, $I(3)=\{(3,1),(3,2)\}$가 된다. 식(18)~(25) 로부터

비용제약식에 대해서 풀면

$U_1(3)=0.583$, $L_1(3)=0.512$, $\alpha_1(3)=(1,1)$

무게제약식에 대해서 풀면

$U_2(3)=0.643$, $L_2(3)=0.512$, $\alpha_2(3)=\{(2,2)\}$

노드의 상한값과 하한값을 구하면

$U(3)=\min\{0.583, 0.643\}=0.583$

$L(3)=\max\{0.512, 0.512\}=0.512$

$\alpha(3)=(1,1)$

$A(3)=\{(1,1),(1,2),(1,3),(1,4),(1,5)\}$

$B(3)=\{(1,1)\}$

단계 4 에서 노드 1 을 제거한 뒤 $U^*=\max\{0.573, 0.583\}=0.583$ $L^*=\max\{0.573, 0.512\}=0.573$ 그리고 $j^*=3$이 결정되었다.

같은 방법으로 위의 절차를 계속하는 과정을 그림 2 로 나타내면 다음과 같다.

최종적으로 노드 2 에서 $U^*=L^*=0.573$이 얻어진다. 참조표를 이용하여 $X_{11}=X_{12}=X_{22}=X_{31}=1$과 나머지는 0 을 얻는다. 위의 결과로 부터 중복 설계의 최적수를 구하면 다음과 같다. 즉, $n_1=1$, $n_2=2$, $n_3=1$ 그리고 결과의 신뢰도는 0.8937 이다.

6. 결 론

이 논문은 다제약식 하에서 중복 최적 문제의 정확한 해를 구하기 위한 새로운 방법을 제시하였다. 이 문제는 일반적으로 비선형정수계획법 (NIP) 문제로 형성되기 때문에 정확한 해를 구한다는 것은 어려운 것이므로 대부분의 학자들이 근사해를 얻으려 하였다.

이 논문에서 NIP 문제는 보다 쉽게 조작할 수 있는 선형배낭문제로 재형성할 수 있음을 보여준다. NIP와 LK문제는 일대일 대응 관계이다.

Branch and Straddle 절차는 결과적으로 LK문제를 풀기 위하여 제시되었다. 이 절차의 효율은 일반의 Branch and Bound 알고리즘과는 달리 복수의 변수를 동시에 조작할 수 있다는 것이다. 다른 배낭해법논리로 Branch and Straddle 절차를 대신하여 사용할 수 있다.그러나 예제에서 본 바와 같이 이 절차는 프로그램 작성하기가 간단하고 최적해를 구함에 있어서 바르다는 것이다. 부가적인 장점으로 이 Branch and Straddle 절차는 부산물로 중간해를 제공한다는 것이다.

참 고 문 헌

[1] Aggarwal, K., "Redundancy optimization in General Systems," IEEE Trans. Reliability, Vol. R-25, pp. 330-332.

[2] Bellman, R., and Dreyfus, S., "Dynamic programming and reliability of multicomponents of devices," Operations Research, Vol. 6 (1958), pp. 200-206.

[3- Greenberg, H., and Hegerich, R., "A branch search algorithm for the knapsack problem," *Mgmt Sci.*, Vol. 16 (1970), pp. 327-333.

[4] Hyun, K.N., "Reliability optimization by 0-1 programming for a system with

several failure modes," *IEEE Trans. Reliab.*, Vol R-24 (1975), pp. 206-210.

[5] Kolesar, P.J., "A branch and bound algorithm for the knopsack problem," *Mgmt Sci.*, Vol 13 (1967), pp. 723-735.

[6] Luus, R., "Optimization of system reliability by a new nonlinear integer programming procedure," *IEEE Trans. Reliab.*, Vol R-24 (1975), pp. 14-16.

[7] McLeavy, D.W., "Numerical investigation of optimal parallel redundancy in series systems," *opl. Res.* Vol 22 (1974), pp. 1110-1117.

[8] Misra, K.B., "Optimum reliability design of a system containing mixed redundancies," *IEEE Trans. PAS,* Vol. 94, (1975), pp. 983-993.

[9] Moskowiz, F., and McLeavy, J., "Some reliability aspects of system design," IRE Trans. Reliability and Quality Control, Vol. PGRQC-8, pp. 7-35.

[10] Nakagawa, Y., Nakashima, K., and Hattori, Y., "Optimal reliability allocation by branch and bound technique," IEEE Trans. Reliability, Vol. R-27 (1978), pp. 31-38.

[11] Nakagawa, Y., and Nakashima, "A heuristic method for determining optimal reliability allocation," IEEE Trans. Reliability, Vol. R-26 (1977), pp. 156-161.

[12] Salkin, H.H., and Dekluyver, C.A., "The knapsack problem: a survey," Naval Logistics Quartery, Vol. 22 (1975), pp. 127-144.

[13] Shih, W., "A branch and bound method for the multiconstraint zero one knapsack problem," Journal of Operational research society, Vol. 30 (1979), pp. 369-378.

[14] Tillman, F.A., "Optimization by integer programming of constrained reliability problems with several modes of failures," IEEE Trans. reliability, Vol. R-18 (1969), pp. 47-53.

[15] Tillman, F.A., Hwang, C.L., and Kuo, W., "Optimization techniques for system reliability with redundancy—a review," IEEE Trans. Reliability, Vol. R-26 (1977), pp. 148-155.

[16] Yun, D.K., and Park, K.S., "Redundancy optimization by linear knapsack approach," International Journal of System Science, Vol. 13 (1982), pp. 839-848.