이산설계공간에서 직교배열표를 이용한 순차적 알고리듬의 국부해

이 정 욱* · 박 경 진†

(2004년 4월 23일 접수, 2004년 7월 7일 심사완료)

Local Solution of a Sequential Algorithm Using Orthogonal Arrays in a Discrete Design Space

Jeong-Wook Yi and Gyung-Jin Park

Key Words : Discrete Design Space(이산설계공간), Orthogonal Arrays(직교배열표), Design of Experiments(실험계획법), Local Solution(국부해), Structural Optimization(구조최적설계)

Abstract

Structural optimization has been carried out in continuous design space or in discrete design space. Generally, available designs are discrete in design practice. However, the methods for discrete variables are extremely expensive in computational cost. An iterative optimization algorithm is proposed for design in a discrete space, which is called a sequential algorithm using orthogonal arrays (SOA). We demonstrate verifying the fact that a local optimum solution can be obtained from the process with this algorithm. The local optimum solution is defined in a discrete design space. Then the search space, which is a set of candidate values of each design variables formed by the neighborhood of a current design point, is defined. It is verified that a local optimum solution can be found by sequentially moving the search space. The SOA algorithm has been applied to problems such as truss type structures. Then it is confirmed that a local solution can be obtained by using the SOA algorithm

1. 서 론

최적설계는 설계의 주요 성능 인자들과 요구사항을 목적함수와 구속조건으로 정식화하여, 구속조건을 만족하면서 목적함수를 최소화하는 자동화된 설계 기법이다.[1~3] 특히 구조설계 분야에서는 유한요소법의 발달과 컴퓨터의 성능향상으로 최적설계의 적용이 활발히 진행되고 있다.[4,5]

최적설계는 설계변수의 종류에 따라 연속설계변수와 이산설계변수로 분류할 수 있다. 연속설계변수를 갖는 문제의 최적화에 대해서는 함수의 구배(gradient)를 이용한 많은 연구가 이루어져 효율적인 알고리즘이 개발되어 있으나, 이산설계변수를 갖는 문제는 연속설계변수를 갖는 문제의 최적화 방법을 그대로 적용할 수 없다.[6~8] 그러나 실제 구조물의 최적설계에서는 설계변수가 어떤 특정한 값들 중에서 선택해야 하거나, 이미 규격화되어 있는 부품의 치수를 취급해야 하는 경우가 많다. 그러므로 이산설계변수를 갖는 문제를 처리하기 위한 방법이 필요하다.

이산설계 최적화를 위한 여러 가지 알고리즘이 제안된 바 있다. Land와 Doig[9]는 선형계획 영역에서 고전적인 분단탐색법(branch and bound) 방법을 개발하면서 이산최적화의 분기점을 마련하였고, Salajegheh[10] 등은 분단탐색법을 정수계획법(integer programming)을 위해 전개하였다. Dakin[11]은 이를

* 한양대학교 최적설계신기술센터
E-mail : yijwook@ihanyang.ac.kr
† 책임저자, 회원, 한양대학교 기계정보경영학부
E-mail : gjpark@hanyang.ac.kr
TEL : (031)400-5246 FAX : (031)407-0755

한 차원 높여 비선형 계획 영역에서 분단탐색법을 제시하였다. Kirkpatrick[12] 등은 조합적 최적화 문제의 한 해법으로 시뮬레이티드 어닐링(simulated annealing)을 제안하였다. 이는 Gidas,[13] Mitra,[14] Hajek[15]에 의하여 전역 최소점에로의 수렴성이 이론적으로 증명되었고, 기본개념의 단순성과 범용성이 두드러져 특별한 대안이 없을 때 편리하게 사용할 수 있다. 그러나 실제 문제에 적용시, 초기치 설정, 변동량, 수락기준 등 많은 문제점들을 가지고 있다. 유전 알고리듬(genetic algorithm)은 유전학과 자연진화를 흉내낸 적응탐색법으로 Holland[16]가 처음 소개하였다. 유전 알고리듬을 전통적인 조합최적화 문제에 적용하면서 초기에 사용하였던 이진표현의 한계성을 알고 이를 개선하기 위한 표현법을 다양화하였다. 특히, Michalewicz[17]는 특수 연산자, 부적합한 해를 복구해주는 복구 알고리듬에 대한 연구를 수행하였고, Goldberg[18]는 해의 정밀도 개선과 구속조건을 쉽게 다룰 수 있도록 실수 코딩 염색체를 사용하는 방법을 제안하였다. 또한 Kido[19] 등은 타부 탐색법(taboo search), 시뮬레이티드 어닐링의 다른 탐색 알고리듬과의 결합을 시도하였다. 이들 알고리듬은 최근까지 많은 연구가 이루어져 효율성이 향상되었으나, 아직 함수계산 횟수가 많다는 단점이 있다.

실험계획법을 이용한 최적화 방법에 관한 연구도 이루어지고 있다. 이권희[20] 등은 연속설계공간에서 구조물의 최적화를 수행한 후, 후처리 단계에서 다구찌법(Taguchi method)에 의하여 이산설계공간에서 최적점을 구하는 방법을 제안하였다. 이권희, 박경진[21]은 다구찌법을 이용하여 제한조건이 없는 문제와 제한조건이 있는 문제에 대한 강건최적설계를 수행하였다. Rao[22]는 목적함수를 다구찌법의 특성함수로 정의하고 설계변수의 영역을 줄여가면서 반복적으로 최적의 해를 구하는 알고리듬을 제안하였다.

최근 이정욱[23] 등은 직교배열표에 의한 행렬실험을 반복적으로 수행하여, 함수계산 횟수를 효율적으로 줄이면서 최적의 해를 찾는 알고리듬을 제안하였다. 그러나 제안한 알고리즘을 통하여 구한 해가 국부해를 찾을 수 있는가에 대한 의문을 안고 있었다. 이에 이산설계공간에서의 국부해를 정의하고, 순차적 알고리듬이 국부해를 찾을 수 있음을 증명한다. 그리고 직교배열표를 이용한 순차적 알고리듬이 국부해를 구할 수 있도록 수정된 알고리듬을 제안한다. 또한 알고리듬의 효율을 높이기 위한 방안을 제시한다. 이를 통하여 이산설계공간에서 직교배열표를 이용한 순차적인 접근방법이 국부해를 찾을 수 있음에 타당성을 부여하고자 한다.

2. 이산설계공간에서의 국부해

2.1 전역 최적화

최적설계는 부과된 조건을 만족하면서 설계자가 원하는 특성치를 극대화시키는 설계변수를 결정하는 방법이다. 설계자가 원하는 특성치를 목적함수 $f(\mathbf{b})$, 부과된 조건을 등제한조건 $h(\mathbf{b})$ 와, 부등제한조건 $g(\mathbf{b})$ 로 정의할 때 이산설계공간에서의 최적설계를 위한 정식화는 식 (1)과 같이 나타낼 수 있다.

$$\begin{aligned} &\textbf{Find} && \mathbf{b} \\ &\textbf{to minimize} && f(\mathbf{b}) \\ &\textbf{subject to} && h_i(\mathbf{b}) = 0, \quad i = 1, \cdots, l \\ & && g_j(\mathbf{b}) \le 0, \quad j = 1, \cdots, m \\ & && \mathbf{b} \in \mathbf{S} \end{aligned} \tag{1}$$

여기서 $\mathbf{b} \in R^n$ 는 이산설계변수이고, $\mathbf{S}$ 는 설계변수가 존재할 수 있는 공간으로 이산설계공간(discrete design space)을 의미한다.

만약 $\mathbf{S}$에서 정의된 어떤 함수 $f(\mathbf{b}^*)$가 $\forall \mathbf{b} \in \mathbf{S}$에 대해 $f(\mathbf{b}^*) \le f(\mathbf{b})$을 만족한다면, 한 점 $\mathbf{b}^* \in \mathbf{S}$는 전역해(global solution)이다. 정의대로 전역해는 이산설계공간에서 주어진 설계변수에 대하여 모든 조합에 대한 함수 값의 계산을 필요로 한다. 예를 들어, 설계변수가 10개이고 이산설계공간 내에 존재하는 범위가 각각 10개씩 존재한다면, 10^{10}번의 함수 값 계산이 필요하다. 설계변수가 증가함에 따라서 함수 값의 계산은 기하급수적으로 증가하기 때문에, 전조합 계산을 통하여 최적의 해를 구하는 방법은 유용하지 않다.

2.2 국부 최적화

이산설계공간 $\mathbf{S}$ 에서 정의된 임의의 설계변수 $\mathbf{b}^* \in \mathbf{S}$를 중심으로 정의된 이산설계공간 $\mathbf{S}^*$가 있다. 여기서 $\mathbf{S}^* \in [\,\mathbf{b}_1 \;\; \mathbf{b}_2 \;\; \mathbf{b}_3 \;\; \cdots \;\; \mathbf{b}_{p-1} \;\; \mathbf{b}_p \;\; \mathbf{b}^*\,]$, $p = 3^n - 1$으로 설계변수 주변의 개수이고, n은 설계변수의 개수이다. Fig. 1은 설계변수 A와 B의 2개를 갖는 이산설계공간 $\mathbf{S}^*$를 표현하고 있다. 만일 $\forall i$ 에 대하여 $f(\mathbf{b}^*) \le f(\mathbf{b}_i)$ 이라면, $\mathbf{b}^*$를 이산공간에서의 국부해(local solution)라 정의한다.

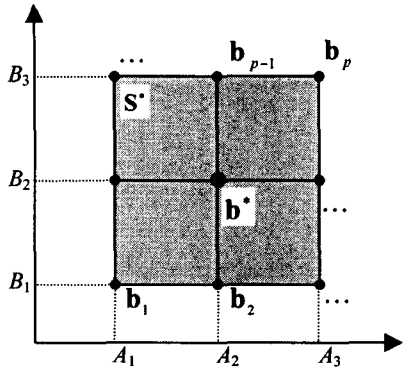

Fig. 1 Discrete design space with two design variables

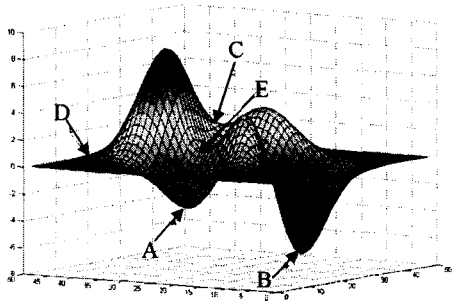

Fig. 2 Global and local solutions in discrete design space

Fig. 2를 통하여 앞 절에서 정의한 전역해와 국부해를 살펴보면, 점 B는 전역해임 알 수 있다. 그러나 전역해를 구하는 것은 비용이 너무 많이 들기 때문에 국부해를 찾는 것이 하나의 방법일 수 있다. 국부적 최적화는 설계변수의 주어진 영역에서 국부지역을 선택하여 목적함수를 최소화하는 해를 찾고, 다시 해의 지역을 중심으로 최소화하는 해를 찾는 반복적인 과정을 통하여 구하는 방법이다. 이 과정을 통하여 구한 최적해는 초기치의 위치에 따라서 점 B일 수 있지만, 점 A를 구할 수도 있다. 또한 경우에 따라서는 점 C, D, E 등을 최적해로 구할 수 있다. 그러나 초기의 특성치에 비하여 성능을 향상시킬 수 있는 해를 구한다면 공학적인 문제에서 유용하게 쓰일 수 있다.

이산설계공간에서의 최적화 과정에서 국부 최적해를 구하는 경우에도 많은 반복적인 과정을 거쳐 구하게 된다. 지역에 이웃하는 영역에 대하여 설계변수의 전조합에 대한 특성함수를 계산해야 하므로, 설계변수의 개수와 초기치의 위치에 따라서 특성함수의 계산량은 급속도로 증가할 수 있다.

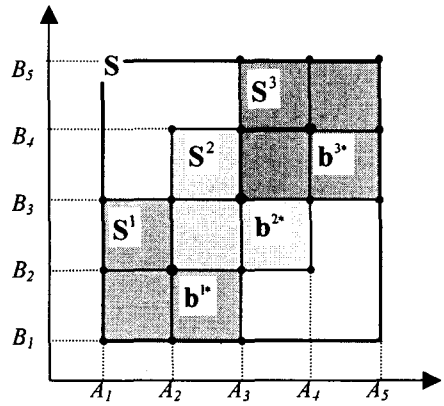

Fig. 3 Search space with two design variables in the discrete space

3. 직교배열표를 이용한 순차적 알고리듬에 의한 국부해

3.1 순차적 알고리듬에 의한 국부해

이산설계공간 $\mathbf{S}$에서 현재의 설계변수를 중심으로 하고, 주변의 설계변수를 배치한 탐색공간(search space) $\mathbf{S}^j$를 정의한다. 설계변수가 2개인 경우, 이산설계변수의 현재 값을 2수준에 배치하고 2수준보다 크거나 작은 주변의 설계변수를 각각 1수준과 3수준에 배치하여 Fig. 3과 같이 탐색공간을 나타낼 수 있다. 여기서 A와 B는 설계변수이고, A_i와 B_i (i = 1,2,3,4,5)의 전조합으로 이루어지는 공간은 이산설계공간 $\mathbf{S}$이다. 또한 $\mathbf{S}^j$와 $\mathbf{b}^{j*}$는 j-번째 탐색공간과 탐색공간에서의 설계변수 벡터이다. j-번째 탐색공간에서 $\mathbf{b}^{j*}$를 중심으로 주변의 설계값은 전조합에 대한 후보값 중에서 선택할 수 있다.

현재의 탐색공간에서 목적함수가 최소인 최소해를 구하고, 최소해를 중심으로 탐색공간을 재정의하여 이동하는 순차적 반복과정을 생각할 수 있다. 2.2절에서 정의한 국부해의 정의에 따르면, j-번째 탐색공간에서 전조합에 대한 함수값을 비교하여 가장 작은 최소해인 $\mathbf{b}^{j*}$를 찾을 수 있다. 이때 $\mathbf{b}^{j*}$는 국부해이다. 만일 j-번째 탐색공간에서 $\mathbf{b}^{j*}$가 국부해가 아니라면, $\mathbf{b}^{j+1*} \neq \mathbf{b}^{j*}$이므로 탐색공간을 이동하고, 이동한 공간에서 $f(\mathbf{b}^{j+1*}) \leq f(\mathbf{b}^{j*})$인 최소해를 찾을 수 있다. 그러나 $\mathbf{b}^{j*}$가 국부해라면, 다음 반복과정에서도 같은 탐색공간으로 이동하므로 반복과정을 종료할 수 있다. 그러므로 탐색공

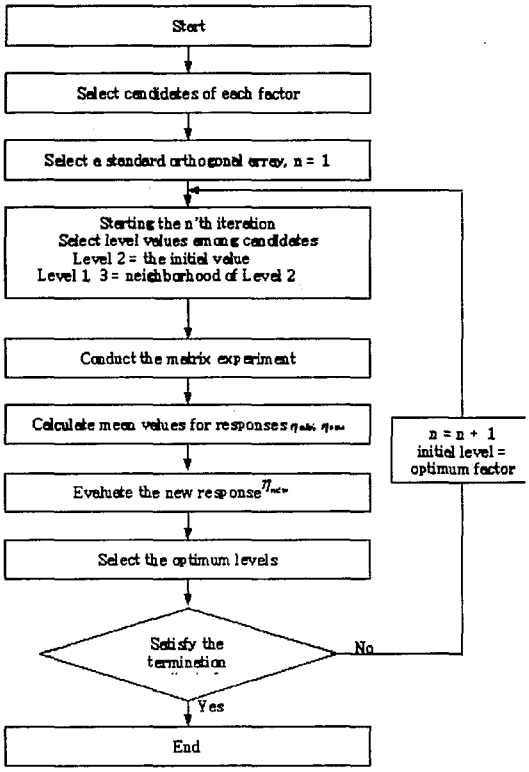

Fig. 4 Flow of discrete design for constrained problems

간을 순차적으로 이동함으로써 국부해를 구할 수 있다.

3.2 직교배열표를 이용한 순차적 알고리듬

앞 절에서는 탐색공간을 순차적으로 이동함으로써 국부해를 구할 수 있음을 알 수 있었다. 여기서는 이산설계공간에서 직교배열표를 이용하여 효율적으로 최적화 과정을 수행하는 알고리듬을 간략히 소개하겠다. 이는 직교배열표를 이용한 순차적 알고리듬(sequential algorithm using orthogonal arrays; SOA)으로 이전에 제안한 바 있다.[23] 제안한 알고리듬은 탐색공간에서의 전조합에 대한 함수계산 횟수를 효율적으로 줄이기 위하여 직교배열표를 이용한다. 만일 교호작용이 존재하지 않는다면, 전조합에 대한 함수계산을 한 것과 동일한 효과를 누릴 수 있으므로 국부해를 찾을 수 있다. 그러나 교호작용이 존재할 시에는 전조합에 대한 추정치와 실제 함수값과의 오차가 존재하여, 현 탐색공간에서의 획득한 해가 최소해가 아닐 수 있다. 그러므로 앞 절에서 정의한 국부해를 찾기 위하여 현 반복과정에서 국부해로 추정되면 전조합에 대한 확인실험을 실시하도록 알고리듬을 수정한다. 이는 현재의 탐색공간에서 찾은 최소해가 국부해인지를 확인할 수 있고, 만일 국부해가 존재하지 않는다면 최소화하는 방향으로 탐색공간을 이동하도록 수정하여 준다. 또한 알고리듬의 효율을 높이기 위한 방안이 있다. 첫번째로 표준직교배열표를 변경하여 사용하는 것이고, 두번째로 최적조건을 탐색할 때 전조합에 대한 추정치를 계산하여 올림차순으로 정리한다는 것이다. Fig. 4에는 제한조건이 있는 문제에 대한 순차적 알고리듬의 흐름을 보이고 있다. 단계별로 수정된 알고리듬을 설명하겠다.

Table 1 Orthogonal arrays, $L_9(3^4)$

Expt. No.	Column number			
	A	*B*	*C*	*D*
1	1→2	1→2	1→2	1→2
2	1→2	2→1	2→1	2→1
3	1→2	3	3	3
4	2→1	1→2	2→1	3
5	2→1	2→1	3	1→2
6	2→1	3	1→2	2→1
7	3	1→2	3	2→1
8	3	2→1	1→2	3
9	3	3	2→1	1→2

1 단계: 문제설정

문제의 특성치(목적함수)와 인자(설계변수)를 설정한다. 각 인자의 후보값은 이산설계공간에서 설계변수의 경계(boundary)를 벗어나지 않는 가능한 영역 내에 있는 이산 값들을 설정한다.

2단계: 직교배열표 선택 및 초기치 설정

행렬실험에서 사용할 3수준 계의 직교배열표를 선택한다. 인자의 개수에 따라서 적절한 표준직교배열표를 선택하는데, Table 1의 예와 같이, 선택한 표준직교배열표의 1수준을 2수준(굵은 글씨)으로 변환하고 2수준을 1수준(굵은 글씨)으로 변환한다. 모든 표준직교배열표는 행렬실험 시 $A_1B_1C_1D_1$ 조건의 실험을 포함하는 반면, $A_2B_2C_2D_2$ 조건의 실험은 포함하지 않는다. SOA 알고리듬에서 최적조건은 전조합에 대한 추정치를 비교하여 구한 최적조건에서의 확인실험 결과와 직교배열표의 행렬실험상의 결과를 비교하여 구한다. 그러므로 현재의 반복과정에서 최적수준을 선정시, 전 반복과정의 설계값에 해당하는 $A_2B_2C_2D_2$ 조건을 직교배열표에 포함시킴으로써 전 반복과정의 결과와 현 반복과정의 결과를 자동적으로 비교할 수 있다.

다음으로 해당 인자의 수준값들을 배치한다. 수준값들은 인자의 후보값들 중에서 선택하여야 하

며, 초기치는 2수준에 배치하고 1과 3수준에는 초기치의 전·후에 위치한 후보값을 배치한다. 한번의 반복과정이 끝나고 탐색공간에서의 최적해를 선정하면, 각 인자에 대하여 최적조건의 후보값을 2수준에 배치하고 인접한 후보값들을 1과 3수준에 배치한다.

3단계: 행렬실험을 실시

2단계에서 수정한 직교배열표의 각 행마다 배치된 인자의 수준에 대한 행렬실험을 실시하고 각 행의 반응치를 구한다. 제한조건이 있는 문제에서는 목적함수에 대한 반응치 η_{obj}와 각 제한조건에 대한 반응치 η_{cons}를 구할 수 있다.

4단계: 현 탐색공간에서의 최적조건 선정

행렬실험을 통해서 나온 결과를 바탕으로 전조합에 대한 목적함수와 제한조건의 추정치 $\hat{\eta}_{obj}$, $\hat{\eta}_{cons}$를 구한다. 제한조건이 있는 문제를 해결하기 위해서 벌칙계수를 포함하는 벌칙함수(penalty function)를 도입하였다. 벌칙함수는 식 (2)에 의하여 구한다.

$$\hat{P} = s \times \sum_{i=1}^{m+l} \max[0, \hat{v}_i] \tag{2}$$

여기서 $\hat{P}$는 추정치의 벌칙함수, $\hat{v}_i$는 i-번째 제한조건의 위배량(maximum violation)의 추정치이고 s는 조절계수(scale factor)이다. 기존의 방법은 행렬실험에서 구한 목적함수와 제한조건의 벌칙함수를 합산한 반응치를 바탕으로 평균분석(analysis of means; ANOM)을 실시하였다. 그러나 평균분석에 의하여 최적조건을 선정 시, 행렬실험의 어느 한 경우라도 제한조건을 위배하는 경우에는 모든 조건의 평균치에 벌칙함수 값이 더해져 왜곡된 결과를 낳는다. 그러므로 정확한 추정치를 구하기 위하여 알고리듬을 수정하였다. 이는 먼저 목적함수와 제한조건의 추정치를 구하고, 전조합에 대한 새로운 반응치를 구하도록 하는 방법이다.

전조합에 대한 새로운 반응치는 식 (3)과 같이 벌칙함수를 포함하며, 각 인자의 수준에 대한 추정치이다.

$$\hat{\eta}_{new} = \hat{\eta}_{obj} + \hat{P} \tag{3}$$

이들을 새로운 반응치의 크기에 따라 올림차순으로 정렬한다. 그러면 최상위에 배치된 조건에 의한 반응치가 최소의 반응치를 갖는다. 그러나 이는 추정치에 의한 결과이므로 실제로는 제한조건을 위배할 수 있다. 그러므로 확인실험을 통하여 실제로 제한조건을 만족하는지 여부를 확인하고, 만족하지 않는다면 다음 순위의 조건에 대하여 확인실험을 행한다. 이와 같이 올림차순의 순서대로 확인실험을 하여 제한조건을 만족하는 최소수준을 구한다. 그러나 교호작용이 존재하는 경우에는 추정치에 의한 순위에 오류가 존재할 수 있다. 그러므로 위의 과정을 통하여 구한 최소수준에서의 새로운 반응치와 직교배열표 상의 행렬실험에 따른 반응치를 비교하여 최적의 수준조합을 갖는 최적수준을 선택한다.

5단계: 종료조건

종료조건은 (1)현재 설계변수의 탐색공간과 최적수준을 중심으로 한 설계변수의 탐색공간이 같은 경우 (2)제한조건을 위배하는 해가 연속적으로 구해지는 반복횟수로 한다. 종료조건을 만족시키지 않는다면 2단계로 돌아가서 같은 과정을 반복적으로 수행한다.

종료조건에서 (1)의 경우는 앞에서 정의한 국부해를 찾은 경우이다. 종료조건에 의한 결과가 국부해임을 증명하기 위해서, 현재의 탐색공간에서 전조합에 대한 반응치를 구하여 비교하도록 알고리듬을 수정하였다. 전조합에 대한 반응치를 구하여 얻은 최적수준이 모든 인자에 대하여 2수준이면 정의에 따라 국부해임을 알 수 있다. 그렇지 않은 경우는 국부해를 구하지 못하였으므로 다시 2단계로 돌아가 반복과정을 진행한다.

(2)의 경우는 모든 행렬실험이 제한조건을 위배하는 경우로 비가용 영역(infeasible region)으로 빠져든 경우이다. 이는 알고리듬이 이웃의 탐색공간을 이동해나가면서 목적함수와 벌칙함수의 합으로 이루어진 새로운 반응치를 강하함수(descent function)를 줄이는 방향으로 나아가기 때문이다. 이때는 강제적으로 반복과정을 중지시키고 지금까지 구한 해 중에서 제한조건을 만족하는 최소해를 선택한다.

지금까지 수정된 순차적 알고리듬을 설명하였다. 이산설계공간에서 순차적 알고리듬을 이용한 최적화는 함수계산 횟수를 줄이면서 국부해를 찾을 수 있는 효율적인 알고리듬이다. 다음 장에서는 알고리듬을 적용하여 예제를 풀고 국부해를 찾을 수 있음을 검증한다.

Table 2 Results of example 1 with SOA

Design variables	Initial values	Optimum levels
b_1	0.0	0.0
b_2	0.0	1.0
b_3	0.0	2.0
b_4	0.0	-1.0
Value of obj. fn.	0.0	-44.0
No. of Iteration	·	4
No. of fn. evaluations	·	41

Table 3 Full factorial experiments at (0.0 1.0 2.0 –1.0)

No.	Design variables				Obj.	New Resp.
	b_1	b_2	b_3	b_4		
1	0.0	1.0	2.0	-1.0	-44.0	-44.0
2	0.0	0.5	2.0	-1.0	-42.25	-42.25
3	-0.5	1.0	2.0	-1.0	-41.25	-41.25
.	.	.	.		.	.
.	.	.	.		.	.
.	.	.	.		.	.
80	0.0	1.5	2.5	-1.5	-53.5	149.6
81	0.5	1.5	2.5	-1.5	-55.75	184.17

4. 예제 적용 및 고찰

4.1 수학 예제

아래의 수학 예제는 목적함수와 제한조건이 비선형 함수이지만 교호작용 항을 포함하지 않고 있다. 문제의 정식화는 식 (4)와 같다.

$$\begin{aligned}
&\textbf{Find} && b_i \\
&\textbf{to minimize} && b_1^2 + b_2^2 + 2b_3^2 + b_4^2 \\
& && -5b_1 - 5b_2 - 21b_3 + 7b_4 \qquad (4) \\
&\textbf{subject to} \\
& && b_1^2 + b_2^2 + b_3^2 + b_4^2 + b_1 - b_2 + b_3 - b_4 - 8 \le 0 \\
& && b_1^2 + 2b_2^2 + b_3^2 + 2b_4^2 - b_1 - b_4 - 10 \le 0 \\
& && 2b_1^2 + b_2^2 + b_3^2 + 2b_1 - b_2 - b_4 - 3 \le 0 \\
&\textbf{candidate values} \\
& b_i \in \{-5.0,-4.5,-4.0,\ldots,4.0,4.5,5.0\}\ (i=1,2,3,4)
\end{aligned}$$

각 설계변수가 가질 수 있는 후보값들은 임의로 20개의 이산값으로 선정하였다. 설계변수의 초기치는 **b** = (0.0, 0.0, 0.0, 0.0)이고 이 때의 함수값은 0이다.

설계변수의 수가 4개이고 수준 수는 3인 점을 감안하여, Table 1의 $L_9(3^4)$ 표준 직교배열표를 선택하였다. 순차적 알고리듬을 적용하여 최적화를 수행한 결과, Table 2와 같이 4회의 반복과정 후 종료조건 (1)에 따라서 설계변수 $\mathbf{b}^* = (0.0, 1.0, 2.0, -1.0)$에서 최소값 -44를 구하였다. 전조합에 대하여 확인실험을 실시하여 올림차순으로 정렬한 결과, Table 3과 같이 국부해임을 알 수 있었다.

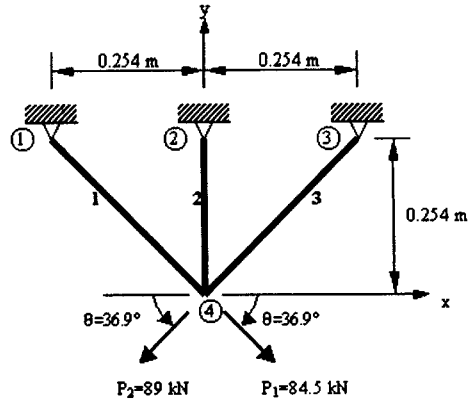

Fig. 5 Three bar truss

4.2 3부재 트러스

Fig. 5와 같이 2개의 하중조건이 3부재 트러스 구조물에 작용할 때, 단면적 A_1, A_2, A_3를 결정하는 문제이다. 물성치는 탄성계수(E) 68.9 GPa, 밀도(ρ) 2770 kg/m^{3}을 사용하였다. 문제의 정식화는 식 (5)와 같다.

$$\begin{aligned}
&\textbf{Find} && A_i \\
&\textbf{to minimize} && \text{mass} \qquad (5) \\
&\textbf{subject to} && -103.4\ \text{MPa} \le \sigma_i \le 137.9\ \text{MPa} \\
&\textbf{candidate values} \\
& A_i\ (\times 10^{-5}\ \text{m}^2) \in \{10, 13, 15, 18, 21, 23, 26, 28, \\
& \quad 31, 34, 36, 39, 41, 44, 46, 49, 52, 54, 57, 59\} \\
& \quad (i=1,2,3)
\end{aligned}$$

여기서 σ_i는 각 부재에 작용하는 응력을 의미한다. 설계변수가 가질 수 있는 후보값들은 임의로 20개의 이산값으로 선정하였다.

설계변수의 수가 3개이고, 수준 수는 3인 점을 감안하여 $L_9(3^4)$ 표준 직교배열표를 선택하였다. 그리고 순차적 알고리듬을 적용하여 최적화를 수행하였다. Table 4와 같이, 5회의 반복과정 후 종료조건 (1)에 따라서 설계변수 A_1, A_2, A_3 = (46 46 49) ($\times 10^{-5}$ m^2)에서 최소값 1.269 kg을 구하였다. 다음으로 국부해를 확인하기 위하여, Table 5와 같이 국부해로 추정한 탐색공간에서 전조합을 실시하여 반응치를 구하였다. 그 결과 순차적 알고리듬에서 추정한 해는 국부해가 아님을 알 수 있었다. 실제로 순차적 알고리듬의 결과보다 더 좋은 A_1, A_2, A_3 = (46 44 49) ($\times 10^{-5}$ m^2)에서 최소값 1.255 kg이 존재

Table 4 Results of example 2 with SOA

Design variables ($\times 10^{-5}m^2$)	Initial	1st SOA	Full Fac.	2nd SOA	Full Fac.
A_1	57	46	46	46	46
A_2	57	46	44	44	44
A_3	57	49	49	49	49
Mass (kg)	1.54	1.269	1.255	1.255	1.255
No. of Iteration	·	5	·	1	·
No. of fn. evals.	1	50	27	10	27

Table 5 Full factorial experiments at (46 46 49)

No.	Design variables ($\times 10^{-5}m^2$)			Mass (kg)	New Resp.
	A_1	A_2	A_3		
1	46	44	49	1.255	1.255
2	46	46	49	1.269	1.269
3	44	49	49	1.270	1.270
·	·	·	·	·	·
·	·	·	·	·	·
·	·	·	·	·	·
26	44	46	46	1.219	68.419
27	44	44	46	1.205	80.939

하였다. 이는 문제의 함수에 교호작용이 존재하기 때문이다. 그러나 순차적 알고리듬에 의한 최소해는 탐색공간에서 전조합에 의한 최소해와 비교하여 비교적 적은 차이를 보이고 있다. 다음으로 전조합을 실시하여 구한 최소해를 중심으로 탐색공간을 설정하고, 순차적 알고리듬에 따른 반복과정을 재진행 하였다. 1회의 반복과정 후, 종료조건 (1)에 따라서 종료하였고, 순차적 알고리듬에서 구한 국부해를 중심으로 한 탐색공간을 설정하였다. 그리고 전조합에 대한 반응치를 구하여 비교한 결과 국부해임을 알 수 있었다. Table 4에서는 최적화를 실시한 결과를 보이고 있다. 최적화 결과 질량을 1.54 kg에서 1.255 kg으로 줄일 수 있었다.

위의 예제는 교호작용이 존재하는 문제로 초기에는 순차적 알고리듬의 반복과정으로 국부해를 구하지 못하였으나, 전조합을 통하여 찾은 해를 중심으로 탐색공간을 재설정하여 반복과정을 재진행한 결과 국부해를 찾을 수 있었다. 1회의 반복과정 중에 필요한 함수계산 횟수는 총 51회였으나, 국부해를 확인하기 위한 2회의 전조합 계산을 포함하면 총 115회로 증가하였다. 그러나 전조합에 대한 확인을 행하지 않았을 때의 최적해와 전조합에 대한 확인을 행한 국부해와의 차이가 크지 않으므로, 전조합에 대한 확인을 행하지 않는 순차적 알고리듬의 결과를 이용하는 것도 효율면에서 타당하다고 생각한다.

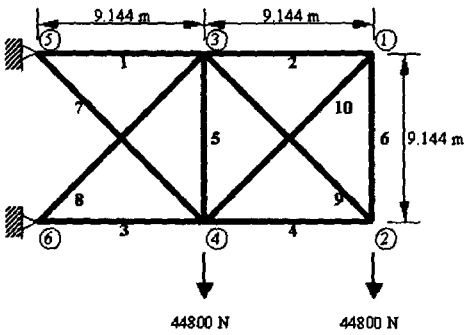

Fig. 6 Ten bar truss

Table 6 Results of example 3 with SOA

Design variables ($\times 10^{-5}m^2$)	Initial	1st SOA	Full Fac.	2nd SOA	Full Fac.
A_1	729	542	542	542	542
A_2	729	194	97	6	6
A_3	729	542	542	542	542
A_4	465	323	323	323	323
A_5	465	97	6	6	6
A_6	465	97	6	6	6
A_7	465	400	400	400	400
A_8	465	400	400	400	400
A_9	465	323	400	400	400
A_{10}	465	97	6	6	6
Mass (kg)	1551.24	879.03	804.42	781.69	781.69
No. of Iteration	·	6	·	2	·
No. of fn. evals.	1	168	59049	56	59049

4.3 10부재 트러스

Fig. 6과 같이 2개의 하중이 10부재 트러스 구조물에 작용할 때 각 부재의 단면적 $A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}$을 결정하는 문제이다. 물성치는 탄성계수($E$) 68.9 GPa, 밀도($\rho$) 2770 kg/m^{3}을 사용하였다. 문제의 정식화는 식 (6)과 같다.

$$\begin{aligned} &\textbf{Find} && A_i \\ &\textbf{to minimize} && \text{mass} \\ &\textbf{subject to} && -172.4\text{MPa} \le \sigma_i \le 172.4\text{MPa} \quad (6) \\ &\textbf{candidate values} \\ &A_i\,(\times 10^{-5}\,\text{m}^2) \in \{6, 97, 194, 323, 400, 465, 542, \\ &\qquad 645, 729, 813\} \quad (i = 1,2,\cdots,10) \end{aligned}$$

여기서 σ_i는 각 부재에 작용하는 응력을 의미한다. 설계변수가 가질 수 있는 후보값들은 임의로 10개의 이산값으로 선정하였다.

설계변수의 수가 10개이고, 수준 수는 3인 점을 감안하여 $L_{27}(3^{13})$ 표준 직교배열표를 선택하여 행렬실험을 실시하고 반복과정을 실시하였다. 순차

적 알고리듬을 적용하여 최적화를 수행한 결과는 Table 6과 같고, 6회의 반복과정을 중 총 169회의 함수계산 후에 종료하였다. Table 6을 보면 전조합에 대한 확인실험을 실시한 결과 순차적 알고리듬으로 구한 해가 국부해가 아님을 알 수 있었다. 다음으로 전조합을 통하여 구한 해를 중심으로 순차적인 반복과정을 실시하였다. 그 결과 Table 6과 같이 종료조건 (1)에 의하여 종료하였다. 국부해임을 확인하기 위하여 국부해로 추정되는 새로운 탐색공간에서 전조합에 대한 확인실험을 실시한 결과 국부해임을 알 수 있었다.

전조합에 대한 확인실험을 실시한 경우에는 국부해를 구할 수 있으나, Table 6과 같이 함수계산 횟수가 기하급수적으로 증가함을 알 수 있다. 그러므로 실제 현장에서 이용하기에는 불가능하다. 그러나 전조합에 대한 확인실험을 실시하지 않고 순차적 알고리듬(SOA)를 1회 실시한 결과를 이용하는 것도 효율면에서 타당하다고 생각한다.

5. 결 론

이산설계공간에서 직교배열표를 이용한 순차적 알고리듬(SOA)에 대한 국부해의 존재를 증명하였다. 먼저 이산설계공간에서 국부해를 정의하였다. 목적함수와 제한조건에 교호작용이 존재하지 않으면 국부해를 찾을 수 있으나, 교호작용이 존재하는 경우 오차가 있을 수 있다. 그러므로 정의한 국부해를 찾기 위하여 순차적 알고리듬(SOA)을 수정하였다. 직교배열표를 이용한 반복과정을 거쳐서 구한 국부해를 확인하기 위하여 국부해를 중심으로 한 탐색공간에서 전조합에 대하여 확인실험을 실시한 후 최적해임을 확인하는 방법이다. 두번째로 순차적 알고리듬의 효율을 높이기 위하여 표준직교배열표의 수준을 변화하여 사용하는 방법을 제안하였다. 이를 통하여 탐색공간의 중심을 자연스럽게 비교할 수 있었다. 세번째로 제한조건이 있는 문제에 대해서는 인자의 각 수준조합에 대한 추정치를 구하는 과정에서 목적함수의 추정치와 각 제한조건의 추정치를 계산한 후에 벌칙함수를 포함한 새로운 반응치를 계산할 것을 제안하였다. 이는 새로운 반응치와 실제 값과의 오차를 줄일 수 있다. 다음으로 수정한 알고리듬을 적용하여, 교호작용이 없는 수학예제와 교호작용이 존재하는 예제를 풀어보았다. 교호작용이 없는 예제에서는 전조합에 대한 확인실험이 없이 국부해를 찾을 수 있었다. 그러나 교호작용이 있는 예제에서는 전조합에 대한 확인실험을 행하여 국부해를 찾을 수 있었다. 예제를 통하여 수정된 알고리듬으로 국부해를 찾을 수 있음을 알 수 있었으나 실제 현장에서 이용 시에는 실험횟수의 기하급수적인 증가로 전조합에 대한 확인실험이 어려울 수 있다. 그러나 순차적 알고리듬을 이용하여 근사적인 해를 구하는 것도 유용성면에서 타당함을 알 수 있었다.

후 기

이 연구는 한국과학재단지정 최적설계신기술센터의 연구비 지원으로 수행되었습니다.

참고문헌

(1) Arora, J.S., 1989, *Introduction to Optimum Design*, McGraw-Hill, New York.

(2) Haug, E.J. and Arora, J.S., 1979, *Applied Optimal Design*, John Wiley & Sons, Inc., New York.

(3) Vanderplaats, G.N., 1984, *Numerical Optimization Techniques for Engineering Design*, McGraw-Hill, New York.

(4) Haftka, R.T., Gurdal, Z. and Kamat, M., 1990, *Elements of Structural Optimization*, Kluwer Academic Publishers, Dordrecht.

(5) Kirsch, U., 1981, *Optimum Structural Design*, McGraw-Hill, New York.

(6) Arora, J.S. and Huang, M.W., 1994, "Methods for Optimization of Nonlinear Problems with Discrete Variables: a review," *Structural Optimization*, Vol. 8, pp. 69~85.

(7) Huang, M.W. and Arora, J.S., 1997, "Optimal Design with Discrete Variables: Some Numerical Experiments," *International Journal for Numerical Methods Engineering*, Vol. 40, pp. 165~188.

(8) Thanedar, P.B. and Vanderplaats, G.N., 1994, "A Survey of Discrete Variable Optimization for Structural Design," *Journal of Structural Engineering, ASCE*, Vol. 121, pp. 301~306.

(9) Land, A.H. and Doig, A., 1960, "An Automatic Method of Solving Discrete Programming Problems," *Econometrica*, Vol. 28, No. 4, pp. 297~520.

(10) Salajegheh, E. and Vanderplaats, G.N., 1993, "Optimum Design of Structures with Discrete Sizing and Shape Variables," *Structural Optimization*, Vol. 6, No. 2, pp. 79~85.

(11) Dakin, R.J., 1965, "A Tree Search Algorithm for Mixed Integer Programming Problems," *Computer Journal*, Vol. 8, No. 1, pp. 250~255.

(12) Kirkpatrick, S., Gelatt, C.D. and M.P. Vecchi, 1983, "Optimization by Simulated Annealing," *Science*, Vol. 220, pp. 671~680.

(13) Gidas, B., 1985, "Non-stationary Markov Chains and Convergence of the Annealing Algorithms," *Journal of Statistical Physics*, Vol. 39, pp. 73~131.

(14) Mitra, D., Romeo, F. and Sangiovanni-Vincentelli, A.L., 1986, "Convergence and Finite-time Behavior of

Simulated Annealing," *Advances in applied Probability*, Vol. 18, pp. 747~771.
(15) Hajek, B., 1998, "Cooling Schedules for Optimal Annealing," *Mathematics of Operations Research*, Vol. 13, pp. 311~329.
(16) Holland, J.H., 1975, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Michigan.
(17) Michalewicz, Z. and Janikow, C.Z., 1991, "Handling Constraints in Genetic Algorithms," *Proceeding 4th International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, CA., pp. 151~157.
(18) Goldberg, D.E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, New York.
(19) Kido, T., Kitano, H. and Nakanishi, M., 1993, "A Hybrid Search for Genetic Algorithms: Combining Genetic Algorithms, Tabu Search, and Simulated Annealing," *Proceeding 5th International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, CA.
(20) Lee, K.H., Eom, I.S., Park, G.J. and Lee, W.I., 1996, "Robust Design for Unconstrained Optimization Problems Using the Taguchi Method," *AIAA*, Vol. 34, No. 5, pp. 1059~1063.
(21) Lee, K.H. and Park, G.J., 2002, "Robust Optimization in Discrete Design Space for Constrained Problems," *AIAA*, Vol. 40, No. 4, pp. 774~780.
(22) Ku, K.J., Rao, S.S. and Chen, L., 1998, "Taguchi-Aided Search Method for Design Optimization of Engineering Systems," *Engineering Optimization*, Vol. 30, pp. 1~23.
(23) Yi, J.W., Park, J.S., Lee, K.H. and Park, G.J., 2001, "Development of an Optimization Algorithm Using Orthogonal Arrays in Discrete Design Space," *Transactions of the KSME A*, Vol. 25, No. 10, pp. 1621~1626.