대한기계학회 2000년도 춘계학술대회논문집 A pp. 725~730 KSME 00S124

구속조건의 효율적인 처리를 위한 유전자 알고리즘의 개발

조영석* • 최동훈**

Development of Genetic Algorithms for Efficient Constraints Handling

Young-Suk Cho and Dong-Hoon Choi

Key Words : Genetic Algorithms(유전자 알고리즘), Optimization(최적화), Global Optimum(전역최적해), Ranking Penalty Method(순위 벌칙방법), Dynamic Mutation Rate(동적변이율)

Abstract

Genetic algorithms based on the theory of natural selection, have been applied to many different fields, and have proven to be relatively robust means to search for global optimum and handle discontinuous or even discrete data. Genetic algorithms are widely used for unconstrained optimization problems. However, their application to constrained optimization problems remains unsettled. The most prevalent technique for coping with infeasible solutions is to penalize a population member for constraint violation. But, the weighting of a penalty for a particular problem constraint is usually determined in the heuristic way. Therefore this paper proposes, the effective technique for handling constraints, the ranking penalty method and hybrid genetic algorithms. And this paper proposes dynamic mutation rate to maintain the diversity in population. The effectiveness of the proposed algorithm is tested on several test problems and results are discussed.

1. 서 론

최근의 여러 분야에서 수학적인 프로그래밍 기법에 의한 최적설계 방법은 점차 널리 사용되어지고 있다. 그런데 실제의 공학문제에서는 해석시 노이즈가 존재하거나 함수의 불연속성으로 인하여 민감도 정보를 얻을 수가 없는 경우, 기존의 최적화 방법은 적용할 수가 없다.

이에 근사화 기법 등이 개발되었으나, 근사 함수의 정확성에 따른 문제점들로 범용적인 적용엔 문제가 있다. 또한, 이산변수를 가지고 있는 문제에서 기존의 선탐에 근거한 최적화 방법은 설계변수를 연속변수로 취하여 해를 구한 뒤, 얻어진 해를 그에 상응하는 이산변수로 치환하여 해를 얻지만 비효율적이다. 그리고 설계영역의 비오 성(non-convexity)으로 인한 국부최적해가 많은 문제에서는 기존의 방법으로 전역최적해를 구하는 것이 어렵다.

이러한 기존의 최적설계 기법의 어려움을 극복하기 위하여 1975 년 Holland 에 의해 제안된 유전자 알고리즘[1~3]은 그 적용의 용이함과 위에서 열거한 기존의 최적화 방법의 단점을 해결할 수 있는 장점으로 공학전반에서 널리 사용되고 있다.

처음에 유전자 알고리즘은 구속조건이 없는 문제를 다루기 위해서 만들어 졌기 때문에, 구속조건을 처리하기 위한 여러 가지 방법이 연구되어져 왔다. 그 중 하나가 진화과정 중 발생한 비가용영역의 해는 모집단내에서 무조건 버리는 방법이나 비가용해가 너무 자주 발생하면 유전자 알고리즘은 구속조건을 만족시키는 몇 개의 해를 찾기 위하여 많은 시간을 낭비하게 되고, 실제의 전역최적값에 가까이 있는 해일지라도 구속조건을 위배하면 무조건 버리기 때문에 알고리즘의 탐 과정을 위해서는 바람직 하지 않다. 다른 방법으로는 구속조건을 유전 연산자에서 고려하여 가용성을 유지하는 방법[3]이나 선형 구속조건에만 국한되어있으므로, 실제적인 적용에는 한계가 있다. 이

* 한양대학교 대학원 기계설계학과
** 한양대학교 기계공학부

에 대부분의 경우 유전자 알고리즘은 구속조건을 처리하기 위하여 벌칙함수 방법을 사용한다. 하지만 이 방법은 유전자 알고리즘이 구속조건의 수가 많아지거나 비선형성이 강하게 되면 구속조건을 만족시키기가 어려워 그 만큼 함수의 계산시간을 낭비하게 되고 벌칙계수를 정하는 것이 문제의 특성에 의존적이어서 이 벌칙계수를 선정하는 것이 어려운 점이 된다.

이에 본 논문에서는 순위 벌칙 방법을 제안하였으며, 기존의 민감도를 이용하는 최적화 방법과의 적절한 결합을 통하여 유전자 알고리즘에서 구속조건을 효율적으로 처리하고자 하였다.

그리고 유전자 알고리즘은 모집단 크기, 교배율, 변이율과 같은 내부 파라미터에 의한 영향을 많이 받으며 이중 변이율은 전역 탐 과정을 위하여 중요한 파라미터이나 이 값의 선정은 최적화 과정 초기에 설계자의 직관에 의하여 고정된 값으로 정해주게 된다. 따라서 유전자 알고리즘에서 문제에 맞는 적절한 변이율을 선정하는 것이 중요하며, 본 논문에서는 이 값을 자동으로 선정하여 알고리즘의 진화과정 중 동적으로 변화시켜 알고리즘의 다양성을 고려하여 탐과정을 하게 하였다.

그리고 본 논문에서 개발한 유전자 알고리즘의 유용성을 검증하기 위하여 몇 개의 최적설계 예제를 수행하였으며 그 결과를 비교 고찰하였다.

2. 유전자 알고리즘

2.1 변수의 표현방법

유전자 알고리즘에서 변수의 표현방법은 실제의 변수를 2 진수의 스트링으로 변환 (인코딩)하여 교배, 변이의 과정을 거치고 다시 실제의 변수로 변환 (디코딩)하여 선택과정을 거치는 2 진수 표현방법[1]과 실제의 변수를 변환과정 없이 직접 유전연산과정을 거치는 실수 표현방법[3]이 있다. 그러나 2 진수 표현은 다차원 고정밀 수치문제에 있어서는 스트링의 길이가 엄청나게 증가하여 적절하지 못하다. 이에 반하여 실수 표현방법은 변수를 2 진수로 기호화 하지 않고 실수 그 자체를 직접 다루는 방법으로 다차원 고정밀 수치문제와 같이 실제의 문제에 있어서 그 응용성이 높다. 따라서 본 논문에서는 실수 표현방법의 유전자 알고리즘을 사용하였다.

2.2 유전 연산과정

유전자 알고리즘에서의 연산과정은 선택, 교배, 변이의 과정이 있으며 주어진 세대동안 이 과정을 반복하여 원하는 최적해를 탐하게 된다.

2.2.1 선택과정

선택 연산자는 잘 적응한 해들은 살아 남고 잘 적응하지 못한 해들은 도태되도록 유도 함으로서 자연 선택현상을 모델링 한다. 일반적으로 널리 사용되는 선택 방법으로는 룰렛 휠 선택법, 토너먼트 선택법, 순위 선택법, 엘리트 보존 선택법이 있으며, 이중에서 엘리트 보존선택법은 세대가 진행되면서 가장 우수한 엘리트의 적응도는 절대로 떨어지지 않는다는 장점이 있다.

2.2.2 교배과정

교배과정은 두 부모개체의 염체를 조합하여 바꿈으로써, 새로운 자손개체의 염체 를 만드는 과정으로 전체 진화 과정중 중요한 역할을 한다. 이에는 다음과 같은 방법이 있다.

● 단순교배

식 (1)과 같이 두 부모개체에서 임의의 한 곳을 교배위치로 하여, 그 이후의 부분을 맞바꾸어 자손개체를 생성한다.

$$\mathbf{X}_1 = [a_1, a_2, \ldots, a_k, a_{k+1}, a_{k+2}, \ldots, a_n]$$
$$\mathbf{X}_2 = [b_1, b_2, \ldots, b_k, b_{k+1}, b_{k+2}, \ldots, b_n]$$

$$\Downarrow$$

$$\mathbf{X}_1' = [a_1, a_2, \ldots, a_k, b_{k+1}, b_{k+2}, \ldots, b_n]$$
$$\mathbf{X}_2' = [b_1, b_2, \ldots, b_k, a_{k+1}, a_{k+2}, \ldots, a_n] \quad (1)$$

위에서 $\mathbf{X}_1$ 과 $\mathbf{X}_2$ 는 모집단내의 개체 (염 체)가 되며, $a_i (i = 1..n)$와 $b_i (i = 1..n)$는 유전자 , 실제의 설계변수가 된다. 여기서 n 은 설계 변수의 개수이다.

● 산술교배

식 (2)와 같이 두 개체 ($\mathbf{X}_1$, $\mathbf{X}_2$)의 일차 결합으로 정의된다.

$$\mathbf{X}_1' = \lambda_1 \mathbf{b} + \lambda_2 \mathbf{a}$$
$$\mathbf{X}_2' = \lambda_1 \mathbf{a} + \lambda_2 \mathbf{b} \quad (2)$$

여기서 λ_1, λ_2의 값으로 고정된 값을 사용하거나, $\lambda_1 + \lambda_2 = 1$ 을 만족하는 임의의 값을 선택한다.

2.2.3 변이과정

초기의 설계 모집단이 가지는 탐공간 의 제한성을 극복하기 위하여, 유전자를 전혀 다른 형태로 변화 시키는 조작이다. 이러한 돌연변이의 과정은 설정된 변이확률에 따라 수행해야 하며 문제에 적절한 변이확률을 선택하는 것이 중요하다.

● 균일변이

임의의 한 유전자를 식 (3)과 같이 전체 설계영역내의 한 점으로 이동 시키는 과정이다.

$$\mathbf{X} = [a_1, a_2, \ldots, a_k, \ldots a_n]$$
$$\Rightarrow \mathbf{X}' = [a_1, a_2, \ldots, a_k', \ldots a_n] \qquad (3)$$

● 비균일변이

이 변이 과정은 식 (4)와 같이 세대수가 증가할수록 선택된 변수의 변이 되는 폭이 적어지므로 초기에는 전체공간에 대한 전역적 탐 을 하고 시간이 흐를수록 탐영역 의 제한을 두어 지역적 탐을 추구한다.

$$a_k' = \begin{cases} a_k + (UB - a_k)\cdot(1 - r^{(1-\frac{t}{T})^b}) \\ a_k - (a_k - LB)\cdot(1 - r^{(1-\frac{t}{T})^b}) \end{cases} \qquad (4)$$

위식에서 UB 와 LB 는 설계변수의 상·하한치를 나타내며, r 은 0 과 1 사이의 랜덤한 값이며 T 는 전체 세대수를, t는 현 세대수를 나타내고 b는 일반적으로 2~5 사이의 값을 사용한다.

3. 효율적인 구속조건의 처리 방법

3.1 순위 벌칙 방법

기존의 유전자 알고리즘은 각 개체에서 적합수의 값과 구속조건의 위배량의 값을 가지고 적합도 함수를 구성 하였으며 구속조건의 위배정도를 다루기 위하여 벌칙계수를 정해 주었다. 하지만, 본 논문에서는 각 개체마다 함수값이 아닌 적 함수의 순위값과 구속조건의 위배량의 합에 의한 순위값의 합으로 구성된 새로운 적합도 함수를 제시하였다. 식 (5)에서 보는 바와 같이 i 번째 개체의 적합도 함수는 기존의 실제 함수의 값으로 구성된 것이 아닌 각 함수의 순위값으로 구성되었다.

$$\text{fitness function}(\Phi_i) = rank(f)_i + rank\left(\sum_{k=1}^{NCON}\max(0, g_k)\right)_i$$
$$i = 1,..,population\ size \qquad (5)$$

Table 1 은 전체 모집단의 크기가 20 일때, 각 개체에서 적 함수와 구속조건에 대한 순위값과 그것의 합으로 구성된 적합도 함수값을 나타내고 있다.

Table 1 Fitness function value by ranking penalty method

Individuals	Objective value	Sum of Const-raint violation value	Fitness function value
1	$Obj_1 = 13$	$Const_1 = 5$	$\phi_1 = 18$
2	$Obj_2 = 9$	$Const_2 = 19$	$\phi_2 = 28$
3	$Obj_3 = 18$	$Const_3 = 2$	$\phi_3 = 20$
.	.	.	.
.	.	.	.
20	$Obj_{20} = 4$	$Const_{20} = 10$	$\phi_{20} = 14$

3.2 민감도를 이용한 방법과의 결합

유전자 알고리즘은 함수의 값만을 사용하여 영역을 탐하 기 때문에 구속조건의 처리에는 어느 정도의 한계가 있다. 이에 본 논문에서는 기존의 민감도 정보를 이용하는 방법과의 적절한 결합으로 구속조건을 좀 더 효율적으로 다룰 수 있다. 여기서 사용된 민감도를 이용한 최적화 방법은 Method of Feasible Directions 이며, 이 방법은 비선형성이 강한 구속조건도 직접 처리할 수 있는 장점[4,6]이 있다. 또한 이 방법은 유전자 알고리즘이 전역탐 능력이 우수한 반면 지역탐능 력이 떨어지는 단점을 보완해 주어 해의 탐능 력을 좀 더 향상시킬 수 있다.

이 방법의 구현은 적합도 함수값의 수렴도를 관찰해 보면 그 값이 갱신되지 않는 정체구간(stationary region)이 오랜 세대동안 존재하는 경우를 볼 수 있으며, 본 알고리즘은 그러한 정체구간이 초기에 사용자가 정해준 세대수 이상이 되면 그 세대의 유전자 알고리즘에서 얻어진 가장 좋은 해를 초기값으로 하여 Method of feasible directions 로 최적화를 수행한다. 이렇게 하면 함수의 기울기가 0 인 설계점을 찾아주며, 얻어진 설계점을 현 세대의 모집단에 포함시키고, 다음 세대의 진화과정으로 넘어가게 된다. 이렇게 제안한 방법은 기존의 방법보다 구속조건을 처리하는 점에서 우수하며 해의 정확성을 향상시켜 준다.

3.3 알고리즘의 다양성을 고려한 변이율

유전자 알고리즘의 성능은 내부 파라미터에 영향을 받으며, 그 중 변이율은 전역 탐을 위해 중요한 파라미터이다. 기존의 유전자 알고리즘은

초기에 설계자의 직관에 의하여 임의의 고정된 값을 주었으나, 제안한 방법은 모집단내 개체들의 분포에 따라 변이율을 변화 시키고자 하였다. , 매 세대마다 모집단 내의 적합도 함수값들의 분산값을 구해서 이전 세대의 분산값보다 크면 다양성의 관점에서 우수하다고 보아 변이율을 줄여 그만큼 임의의 영역을 찾는 확률을 줄여주고, 이전 세대의 분산값보다 작으면 다양성의 관점에서 부족하다고 보아 변이율을 늘려 임의의 영역을 찾는 확률을 늘려주게 된다. 이 방법의 장점은 변이율을 상황에 맞게 자동으로 조정하여 유전자 알고리즘이 미리 선정된 고정된 변이율에 의해서 탐 능력이 저하되는 것을 줄여줌으로써 전체 설계영역내에서 다양성을 유지하면서 영역을 탐하게 된다.

이 과정은 다음과 같은 식 (6)과 (7)에 의하여 변이율을 조정해주게 된다.

case1) $\sigma_p > \sigma_c$

$$\frac{\sigma_p - \sigma_c}{\sigma_p} = K \qquad \begin{array}{ll} K \le 0.25 & : P_r = 0.06 \\ 0.25 < K \le 0.5 & : P_r = 0.07 \\ 0.5 < K \le 0.75 & : P_r = 0.08 \\ 0.75 < K & : P_r = 0.09 \end{array} \qquad (6)$$

위 식에서 σ_p 는 전 세대에서 개체들의 적합도 함수값의 분산값을 나타내며 σ_c 는 현 세대에서 개체들의 적합도 함수값의 분산값을 나타내고, P_r 은 조정된 변이율이다.

case2) $\sigma_c > \sigma_p$

$$\frac{\sigma_c - \sigma_p}{\sigma_c} = K \qquad \begin{array}{ll} K \le 0.25 & : P_r = 0.04 \\ 0.25 < K \le 0.5 & : P_r = 0.03 \\ 0.5 < K \le 0.75 & : P_r = 0.02 \\ 0.75 < K & : P_r = 0.01 \end{array} \qquad (7)$$

4. 제안한 알고리즘의 적용 예

개발된 프로그램의 전체적인 구조는 매 세대에서 각 개체마다 적합수의 값과 구속조건의 위반 정도에 따라 순위를 정하여 적합도 함수를 구성하고 유전 연산과정 후 정해진 식에 의하여 변이율을 조정해 주게 된다. 그리고 적합수값을 체크하여 값이 향상되지 않으면 정체구간으로 간주하여 이러한 정체구간이 사용자가 정해준 값 이상이 되면 Method of feasible directions 를 호출하여 함수의 민감도에 기초한 탐과 정을 한 후, 얻어진 개체를 현 모집단에 포함시킨다. 이러한 과정을 정해준 세대동안 반복하여 원하는 최적해를 탐 해나간다.

제안한 알고리즘의 효용성을 검증하기 위하여 선형·비선형 구속조건을 가지고 있는 3 개의 전역 최적화 문제에 대해 최적설계를 수행하였고 기존의 유전자 알고리즘과 결과를 비교 고찰하였다.

4.1 테스트 예제 1

첫번째 테스트 예제는 설계변수가 6 개이고 4 개의 선형 구속조건과 2 개의 비선형 구속조건을 가지고 있는 최적화 문제로 식 (8)과 같다.

$$\begin{aligned} \textit{Maximize} \quad & f(\mathbf{x}) = 25(x_1-2)^2 + (x_2-2)^2 + (x_3-1)^2 \\ & \quad + (x_4-4)^2 + (x_5-1)^2 + (x_6-4)^2 \\ \textit{subject to} \quad & g(1) = -x_1 - x_2 + 2 \le 0 \\ & g(2) = x_1 + x_2 - 6 \le 0 \\ & g(3) = -x_1 + x_2 - 2 \le 0 \\ & g(4) = x_1 - 3x_2 - 2 \le 0 \\ & g(5) = 4 - (x_3-3)^2 - x_4 \le 0 \\ & g(6) = 4 - (x_5-3)^2 - x_6 \le 0 \\ & 0 \le x_1 \le 5,\ 0 \le x_2 \le 1,\ 1 \le x_3 \le 5, \\ & 0 \le x_4 \le 6,\ 1 \le x_5 \le 5,\ 0 \le x_6 \le 10 \end{aligned} \qquad (8)$$

이 문제의 알려진 전역 최적해는 $\mathbf{x}^*$ = (5.0, 1.0, 5.0, 0.0, 5.0, 10.0) 에서 $f(\mathbf{x}^*) = 310.0$ 이다.

이 문제는 모집단의 크기는 30, 최대 세대수는 1000, 교배율은 0.6, 변이율은 0.05 를 사용하였고 최적화 수행 결과는 Table 2 에 있다. 사용된 연산자는 룰렛 휠 선택을 동일하게 사용하였고 다음과 같이 각각의 경우에 대해 결과를 비교하였다.

경우 1 은 산술교배와 비균일변이 과정을 이용하였고 구속조건의 처리를 위하여 벌칙함수방법을 사용하였다. 경우 2 는 경우 1 과 같은 연산과정을 사용하였고 제안한 구속조건의 처리 방법으로 순위벌칙함수방법과 Method of feasible directions 를 같이 사용하였다. 경우 3 은 경우 2 와 같으나 단순교배와 균일변이과정을 사용하였다. 경우 4 은 경우 2 와 같고 동적변이율의 개념을 사용하였다.

Case 1) 연산자(산술/비균일)+벌칙함수방법
Case 2) 연산자(산술/비균일)+순위벌칙방법+MFD
Case 3) 연산자(단순/균일)+순위벌칙방법+MFD
Case 4) 연산자(산술/비균일)+순위벌칙방법 +MFD+동적변이율

Table 2 Optimization results for test problem 1

	Case 1	Case 2	Case 3	Case 4
x*	4.999998	5.000000	5.000000	4.998739
	0.999914	1.000000	1.000000	0.999737
	4.997692	5.000000	5.000000	4.996265
	0.174246	0.000000	0.000000	0.024309
	4.995263	5.000000	5.000000	4.999178
	9.999047	9.999036	10.00000	9.999736
g(x*)	-3.999912	-4.000000	-4.000000	-3.998477
	-0.000087	0.000000	0.000000	-0.001522
	-6.000084	-6.000000	-6.000000	-5.999002
	0.000256	0.000000	0.000000	-0.000473
	-0.165020	0.000000	0.000000	-0.009384
	-9.980122	-9.999036	-10.00000	-9.996452
f(x*)	308.56854	309.98843	310.0000	309.57804
MFD	293.9482 at x° = (3.0, 0.5, 2.0, 3.0, 2.0, 5.0)			

위의 결과에서 1000 세대 후 경우 1 은 전역최적해를 찾는데 실패하였으나, 제안한 방법을 사용한 경우 2,3,4 는 원하는 최적해에 상당히 근접하였음을 알 수가 있다. 여기서 경우 3 이 실제 최적해를 주었지만, Fig. 1 에서 보면 진화 중반까지 수렴속도가 느리게 향상하였음을 알 수 있다. 마지 에 있는 값은 민감도를 이용한 Method of feasible directions(MFD)를 사용한 경우 주어진 초기값에 의한 국소최적해로 수렴하였음을 알 수가 있다.

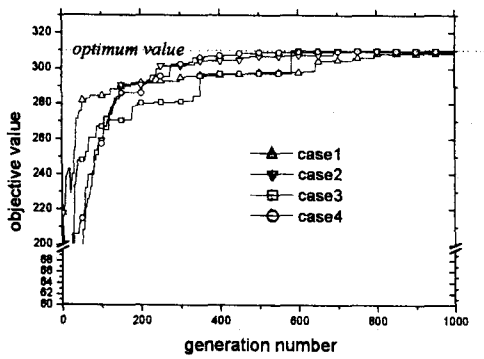

Fig. 1 The convergence history of test problem 1

Fig. 1 은 1000 세대동안의 적합수의 변화를 나타내는 그래프이다. 이 그래프에서 보는 바와 같이 진화 초기에는 경우 1 이 최적해에 빠르게 수렴하나 중반 이후로는 값이 향상되는 정도가 상대적으로 둔함을 볼 수 있다. 경우 3 은 연산자의 선택 때문에 수렴속도가 느림을 알 수가 있고 경우 4 가 300 세대 이후부터 최적해에 빠르게 근접함을 확인할 수 있다.

4.2 테스트 예제 2

두번째 테스트 예제는 설계변수가 7 개이고 4 개의 비선형 구속조건을 가지고 있는 최적화 문제로 식 (9)와 같다.

$$
\begin{aligned}
&\textit{Maximize}\ f(\mathbf{x}) = (x_1-10)^2 + 5(x_2-12)^2 + x_3^4 \\
&\qquad + 3(x_4-11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 \\
&\qquad - 4x_6x_7 - 10x_6 - 8x_7 \\
&\textit{subject to}\ g(1) = 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 - 127 \le 0 \\
&\qquad g(2) = 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 - 282 \le 0 \\
&\qquad g(3) = 23x_1 + x_2^2 + 6x_6^2 - 8x_7 - 196 \le 0 \\
&\qquad g(4) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \le 0 \\
&\qquad -10.0 \le x_i \le 10.0, \quad i = 1,\ldots\ldots,7
\end{aligned}
\tag{9}
$$

이 문제의 알려진 전역 최적해는 **x*** = (2.3305, 1.9514, -0.4775, 4.3657, -0.6245, 1.0381, 1.5942) 에서 f(x*) = 680.6301 이다.

이 문제는 모집단의 크기는 30, 최대 세대수는 3000, 교배율은 0.3, 변이율은 0.05 를 사용하였고 최적화 수행 결과는 Table 3 에 있다. 사용된 연산자는 토너먼트 선택을 동일하게 사용하였고 다음과 같이 각각의 경우에 대해 결과를 비교하였다. 경우 1,2,3,4 는 테스트 예제 1 과 같다.

Table 3 Optimization results for test problem 2

	Case 1	Case 2	Case 3	Case 4
x*	2.038456	2.046243	2.470645	2.336882
	1.644411	2.007348	1.470921	1.984644
	-0.075094	-0.392143	-0.000009	-0.588526
	4.998046	4.275919	5.029062	4.276138
	-0.620945	-0.565081	-0.083535	-0.604586
	1.132240	1.124370	0.461944	1.039605
	1.462791	1.465276	1.818637	1.622632
g(x*)	-0.01101	-0.00005	0.000000	-0.00521
	-257.1221	-255.2754	-255.1801	-251.34353
	-150.4219	-149.0439	-150.2802	-144.80927
	-1.1491	-1.7333	-2.01769	-0.08895
f(x*)	692.1234	682.4199	707.8588	680.87248
MFD	882.4852 at x° = (3.0, -3.0, 2.0, -2.0, 5.0, -5.0, 1.0)			

위의 결과에서 경우 1 보다 경우 2,4 가 더 우수한 값을 주었음을 알 수가 있다. 특히 경우 4 는 얻어진 최적점에서 구속조건의 값을 보면 Method of feasible directions 의 특성으로 구속조건을 좀더 타이트하게 만족시켜 주었고 따라서 실제의 최적해에 상당히 근접하였음을 알 수가 있다.

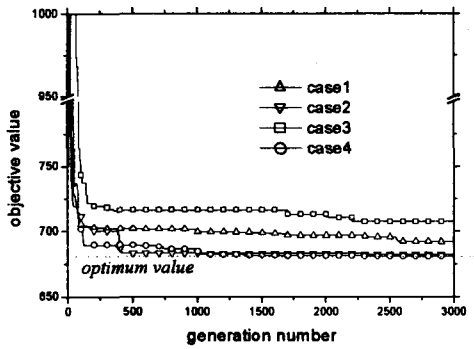

Fig. 2 The convergence history of test problem 2

Fig. 2 은 3000 세대동안의 적함수의 변화를 나타내는 그래프이다. 이 그래프에서 보면 경우 3 은 최적해에 가장 늦게 수렴해 감을 알 수 있고 경우 1 보다는 경우 2 와 특히 경우 4 가 빠르게 해에 수렴해 감을 확인할 수가 있다.

5. 결론

본 논문에서는 유전자 알고리즘에서 구속조건을 효과적으로 처리할 수 있는 방법을 제안하였다. 제안한 첫번째 방법은 순위 벌칙방법으로, 이 방법은 각 개체마다 적 함수의 순위값과 구속조건의 위배량의 합에 의한 순위값의 합으로 적합도 함수를 구성하여 기존의 벌칙함수 방법과는 달리 매 문제마다 벌칙계수를 선정할 필요가 없이 구속조건을 유연하게 다룰 수 있는 장점이 있다. 두번째 방법으로는 함수의 민감도 정보를 이용하는 방법과 유전자 알고리즘을 결합하는 방법을 제안 하였으며, 이 방법은 Method of feasible directions 가 비선형성이 강한 구속조건도 효율적으로 처리해줄 수 있는 장점과 유전자 알고리즘의 지역탐능 력을 향상시킬 수 있는 장점이 있다.

그리고 기존의 방법에서는 전역탐 을 위해 중요한 과정인 변이과정에서 변이율을 초기에 임의로 정하였으며 알고리즘의 성능이 이 값에 민감하게 반응하였다. 그러나 본 논문에서 제안한 방법은 변이율을 따로 정해줄 필요가 없이 모집단 내의 개체들의 분포에 따라 적절한 변이율의 값을 자동으로 선택하기 때문에 모집단내의 다양성을 적절하게 유지하면서 효과적인 전역 탐과정 을 거치게 된다.

제안한 방법은 알고리즘의 해의 정확성과 효율성을 개선하였으므로, 다분야통합최적설계 문제와 같이 다수의 국부 최적해가 존재하는 경우의 전역 최적화를 위하여 보다 유용하게 적용되어 질수 있으리라 기대되어 진다.

후 기

이 연구는 한국과학재단 지정 최적설계신기술 연구센터의 연구비 지원으로 수행되었습니다.

참고문헌

(1) Goldberg, 1997, 'Genetic Algorithms in Search, Optimization, and Machine Learning', Addison Wesley

(2) Gen M., Cheng R., 1996, 'Genetic Algorithms and Engineering Design', Wiley- Interscience

(3) Michalewicz Z., 1996, 'Genetic Algorithms + Data Structures = Evolution Programs', Springer

(4) Vanderplaats G.N., 1984, 'Numerical Optimization Techniques for Engineering Design with Applications', McGraw-Hill

(5) Rao S.S., 1995, 'Engineering Optimization Theory and Practice', Wiley-Interscience

(6) Arora J.S., 1989, 'Introduction to Optimum Design', McGraw-Hill

(7) Stelmack M.A., and Nakashima N., 1998, "Genetic Algorithms for Mixed Discrete/Continuous Optimization in Multidisciplinary Design", AIAA Journal

(8) Okamoto M., and Nonaka T., 1998, "Nonlinear Numerical Optimization with Use of A Hybrid Genetic Algorithm Incorporating the Modified Powell Method", Applied Mathematics and Computation

(9) Sobiesky J.S., Laba K., and Kincaid R., 1998, "Bell-Curved Based Evolutionary Optimization Algorithm", AIAA Journal

(10) Leclerc A.P., B.S., and M.S., 1992, "Efficient and Reliable Global Optimization", Ohio State Univ.