비선형 최적화문제 해결을 위한 혼합유전알고리즘

윤영수*·문치웅**·이상용***

A Hybrid Genetic Algorithm for Solving Nonlinear Optimization Problems

Young-Soo Yun*·Chi-Ung Moon**·Sang-Yong Yi***

요 약

본 연구에서는 비선형 최적화 문제를 효율적으로 해결하기 위한 혼합유전알고리즘(Hybrid Genetic Algorithm : HGA)을 개발하였다.

HGA는 기존 유전알고리즘의 적용에 있어 문제점으로 지적된 정밀도의 적용문제와 벌금함수의 사용을 배제하였으며 지역적최적점으로 빠르게 수렴하는 기존의 지역적 탐색법과 유전알고리즘 적용이후 수렴된 해 주변에 대한 정밀탐색법을 함께 고려하여 설계하였으며 이를 세가지의 비선형 최적화 문제 적용하여 본 논문에서 개발한 HGA의 유효성을 보였다.

keywords : 혼합유전알고리즘(Hybrid Genetic Algorithm), 지역적 탐색법, 정밀탐색법

* 건국대학교 산업공학과 박사과정
** Department of Industrial & System Engineering, Ashikaga Institute of Technology
*** 건국대학교 공과대학 산업공학과

1. 서 론

비선형최적화모델은 기존의 많은 연구들을 통해서 정식화되었고 그 해법 또한 다양하게 개발되어 왔다. 즉 여러가지 제약조건들을 모두 만족해야 하는 설계변수들의 집합중에서 최적해를 구현하는 방법은 기존의 연구들을 통해서 정형화된 기법들이 개발되어져 왔다. 그러나 이러한 최적화 기법들은 탐색과정의 효율은 높혔지만 설계변수들이 정수, 실수, 그리고 이산값을 함께 가지며 제약조건과 목적함수에 서로 혼합되어 사용되는 조합최적화문제(combinational optimal problem)와 여러개의 지역적최적점(local opti-mum)이 존재하는 경우에 전역적최적점(global optimum)을 구하는 문제, 그리고 다목적함수(multi-objective function)의 설계시 최적해 집합의 효율적인 구현 문제들에서는 아직까지 효과적이고 광범위하게 적용될 수 있는 방법이 개발되어 있지 못하다. 따라서 이러한 문제점을 쉽게 해결할 수 있는 대안으로 유전알고리즘(genetic algorithm)에 의한 연구가 활발히 이루어지고 있다.

유전알고리즘은 생태계의 진화원리에 바탕을 둔최적해에 대한 탐색 알고리즘의 하나로 다양한 형태의 문제들에 대해 효과적으로 적용가능하며 전역적 최적점에 가까운 해를 구할 수 있다는 장점이 있지만 기존의 유전알고리즘은 부모세대에서 생성된 개체들을 유전연산자를 사용해 다음세대로 유전하게 되는데, 이럴 경우 그들 부모들보다 적합도가 더 떨어진 개체들이 유전되어 자손 세대의 적합도가 더 낮아질 수도 있으며, 이로 인하여 탐색 도중에 최적해에 도달하지 못하고 지역적 최적점(local optimum)에 미성숙 수렴(premature convergence)되어 전역적 최적해(global optimal solution)를 구하지 못하는 경우도 있다. 또한 제약조건이 있는 목적함수를 제약조건이 없는 목적함수로 만들기 위해 일반적으로 벌금함수(penalty function)를 적용하는데 실제적으로는 어떤 벌금함수를 선택하느냐에 따라 결과에 많은 영향을 미치며, 벌금계수(penalty coefficient)를 결정하는데도 정형화된 방법이 없이 설계자의 경험에 의존하고 있다. 그리고 유전알고리즘의 유전연산자를 통해 최적점 부근에 수렴한 해가 있다면 이에 대해 그 주위를 정밀하게 탐색할 방법이 없다.

또한 최근에는 유전알고리즘의 전역적 탐색특징과 기존에 개발된 지역적 탐색 기법의 빠른 수렴특징을 혼합한 연구가 활발하다. Muhlenbein(1992), Gorges-Schleuter(1990)는 각각 유전알고리즘 수행과정에서 지역적 탐색이 중요한 역할을 한다는 것을 이론적으로 주장했고 이를 실험적으로 증명했으며, Miller등(1993)은 NP-Hard 최적화문제에 대해 유전알고리즘에 지역적 개선 연산자를 포함시킨 모델을 제시했다. Goldberg(1989)는 이진개체에 대한 G-bit개선이라는 개념을 사용하여 지역적 탐색기법과의 조합을 연구하였으며, Davis(1987)는 외부이동연산자를 도입함으로서 최적화 문제를 위한 영역규정기법(domain-specific techniques)을 가진 유전알고리즘의 조합을 제시했고, Ackley(1987)는 유전알고리즘 수행과정에서 교차변이의 역할을 줄이고 지역적 탐색법인 hill-climbing을 첨가한 genetic hill-climbing을 제시했다. Reeves(1994)는 지역적 최적점 주위를 탐색할 수 있는 이웃탐색법(neighborhood search method)을 유전알고리즘에 결합한 모델을 개발했으며, Gen과 Cheng(1996)은 기존의 지역적 탐색기법인 휴리스틱기법과 유전알고리즘을 혼합한 라마르크진화(lamarckian evolution)기법과 memetic알고리즘을 개발했다.

따라서 본 연구에서도 기존의 유전 알고리즘이 가졌던 이러한 문제점들을 개선하며 지역적 최적점으로 빠르게 수렴하는 기존의 지역적 탐색법과 유전알고리즘 수행 이후 수렴한 해 주변에 대한 정밀탐색법을 함께 고려한 혼합유전알고리즘(hybrid genetic algorithms : HGA)을 설계하였다.

본 논문은 총 5장으로 구성되어 있다. 제 2장서는 기존 유전알고리즘을 비선형 최적화 문제에 적용할 때 문제점으로 지적되는 사항들과 개선안을 제시하였으며, 제 3장에서는 HGA의 절차를 단계별로 제시하였다. 제 4장에서는 본 연구에서 개발한 HGA의 유효성을 보이기 위해 세가지의 사례연구를 제시하고 이의 적용결과를 분석하였으며, 마지막 제 5장에서는 본 논문의 의의 및 향후의 연구방향을 제시하였다.

본 논문에서 개발한 HGA는 Windows 환경(Pentium-166Mhz, RAM : 32M)하에서 Visual Basic 4.0을 이용하여 개발하였다.

2. 기존 유전알고리즘의 문제점 및 그 개선안

유전 알고리즘은 인공지능의 한 기법으로서 2차원 이상의 복잡한 탐색공간에서 전역적 최적해를 탐색하는데 아주 효율적이며 유연하다고 증명되어져 왔다. 이러한 유전 알고리즘은 생태계의 자연선택(natural selection)과 적자생존(survival of the fitness)에 근거를 두고 있으며, 새로운 집단(new population)을 형성할 때에 과거의 집단(old population)에서 높은 적합도를 가지는 개체(individual)가 높은 확률을 가지고 새로운 집단으로 유전한다는 것이 그 기본적인 원리이다.(Michalewicz, 1994)

이러한 특징으로 인해 유전알고리즘은 여러분야에 적용되고 있다.(Gen and Cheng, 1996) 특히 해가 발견될 공간이 2차원 이상이고 설계변수가 복잡하며 그 형태가 정수·연속·이산등으로 혼합되어 사용되는 비선형조합최적화문제에서는 기존에 개발된 다른 최적화 기법들에 비해 우수한 결과를 보여주고 있다. 하지만 이러한 유전알고리즘도 그 적용에 있어 다음과 같은 몇몇 문제점이 있다.

2.1 설계변수의 표현방법

유전알고리즘에서 가장 널리 사용되는 설계변수의 표현방법은 비트표현법(bit string representa-tion)으로 생물체의 유전형질을 표현하는 방법과 비슷하기 때문에 대부분의 연구자들에 의해 널리 사용되고 있지만 비트표현법이 아닌 기존의 지역적 최적화 기법과의 연계가 어렵고, 변수형태가 연속설계변수일 경우 변수설계시 정밀도(precision)를 주어서 개체를 생성하기 때문에 이 정밀도를 어떻게 주느냐에 따라 수행속도 및 최적해 도달가능성에 많은 차이가 난다는 단점이 있다.

예를들어 소수점 4째 자리까지의 정밀도를 요구하는 연속설계변수를 구현할 경우 코드화(encoding)와 해석(decoding)식은 각각 아래의 식(1), (2)과 같으며(Wu and Chow, 1995)

$$\lambda \geq \log_2 \left(\frac{x^u - x^l}{\varepsilon} + 1 \right) \tag{1}$$

$$x = x^l + I \left(\frac{x^u - x^l}{2^\lambda - 1} \right) \tag{2}$$

where λ : *string length*

ε: *precision*

x : *design variable*

x^u : *upper bound of* x

x^λ : *lower bound of* x

I : *decimal integer*

만일 $x^u = 90$, $x^\lambda = 50$, $\varepsilon = 0.0001$ 이라면 식(1)에 의해 $\lambda = 19$가 되며, 이에 의해서 랜덤발생 된 이진개체가 1000000000000000001 이라면 $I = 262145$ 이며 식 (2)에 의해 실제 표현되는 연속 설계변수(x)는 70.0001이다.

이러한 코드화와 해석절차는 프로그램 설계과정을 더욱 복잡하게 만들며, 프로그램 실행시 방대한 컴퓨터 메모리와 처리시간을 요구하게 된다는 단점이 있다. 또한 정밀도를 설정하는데 있어서도 어떠한 규정이 없으며 설계자가 문제의 유형에 따라 임의로 주고 있는 실정이다. 따라서 정밀도의 미세한 차이로 인해 구하고자 하는 목적함수의 값에 상당한 영향을 줄 수 있는 최적설계 분야에 있어서는 비트표현법이 부적합하다고 할 수 있다.

이러한 단점으로 인해 본 연구에서는 설계변수의 형태에 어떠한 변형을 가하지 않고 그대로 사용하며 이해하기가 쉽고 정확한 해의 구현이 가능한 실수표현법(floating point representation)을 사용하고자 한다. 이 방법은 특히 설계변수가 많고 복잡하며 탐색공간이 2차원 이상인 최적설계 문제의 경우 비트표현법보다 최적해에 도달할 확률이 크며, 기존에 개발된 지역적 최적화 기법과의 연계가 가능하고 프로그램 설계가 쉽다는 장점이 있기 때문에 정밀도를 요구하는 설계문제의 경우 비트발생보다 더 우수한 해를 구할 수 있다.(Michalewicz, 1994; Michalewicz and Janikow, 1991)

2.2 제약조건의 처리문제

유전알고리즘에서 가장 널리 사용되는 제약조건 처리방법은 벌금함수법(penalty function method)으로 제약조건을 위반하는 정도에 대해 벌금함수법을 적용하여 새로운 집단으로 받아들

이고 있지만 제약조건에 대한 위반이 클수록 벌금함수법을 적용할 확률이 커지게 되며, 또한 위반항에 대한 벌금함수값을 너무 높게 주면 수렴속도가 느려지고 너무 낮게주면 최종해가 나빠질 수도 있다는 단점이 있다.

일반적인 벌금함수법의 적용는 다음과 같다.(Goldberg, 1989)

$$\textit{Minimize } f(x) \tag{3}$$

$$\textit{Subject to } g_i(x) \geq 0 \quad i = 1,2,\ldots,m \tag{4}$$

where x is an m vector

$$\textit{Minimize } f(x) + r\sum_{i=1}^{m} \phi\,[g(x)] \tag{5}$$

where r is a penalty coefficient

ϕ *is a penalty function*

위의 식(3), (4)은 일반적인 최적화문제의 식으로 위반함수법을 적용하면 식 (4)의 제약조건이 식(3)의 목적함수에 첨가되어 식 (5)과 같이 되는데 식 (5)에서는 제약조건에 위배된 값이 실제 프로그램 설계과정에서 벌금함수(ϕ), 벌금계수(γ)의 사용으로 인해 제약조건에 접근하게 되며, 여기에 사용되는 값은 각 제약조건과 목적함수의 형태에 따라 다르며 프로그램 설계자가 경험적으로 정하고 있다. 즉 벌금함수와 벌금계수를 정하는데 일정하게 정해진 규칙이 없다.

이러한 단점을 개선하기 위해 본 연구에서는

변수제약법(variable restriction method)을 사용하고자 한다. 이 기법은 제약조건을 만족하는 해의 공간을 제약하는 방법으로 제약조건을 처리하고 있다. 먼저 하나의 개체가 새롭게 만들어지고, 이 개체가 해석(decoding)되어져서 제약조건의 만족여부를 검사하게 된다. 만일 제약조건을 만족하면 이 개체는 새로운 집단으로 받아들여지고, 제약조건을 만족하지 못하면 탈락되고, 또 다른 새로운 개체가 생성되며 다시 제약조건의 위반 여부를 검사하게 된다.

즉 프로그램 설계시 제약조건을 조건문(*if-condition*)으로 처리하여 그 조건을 만족하지 못하면 탈락시키는 방법을 사용하여 처리하였다. 만일 제약조건이 많고 복잡하며, 해가 발견될 가능공간(feasible space)이 적어 제약조건에 탈락하는 개체가 많이 발생하여 프로그램 수행에 비효율적인 측면이 있을 것이라 생각되지만 이러한 문제는 해가 발견될 가능공간에 대한 개체의 다양한 발생과 컴퓨터 처리속도의 향상으로 해결가능하다. 즉 개체의 재발생은 랜덤발생을 하기 때문에 다양하게 개체를 발생시킬 수 있으며 이렇게 재발생된 개체로 제약조건의 만족여부를 빠른 속도로 검사하기 때문에 해가발견될 가능공간이 적고 제약조건이 복잡하더라도 별로 문제가 되지 않는다.

3. 혼합유전알고리즘(HGA)의 개발

본 연구에서 개발한 HGA의 절차는 최적점 부근까지 비교적 빠르게 수렴하는 기존의 지역적수렴기법인 랜덤탐색·검사법(random searech and test method)과 수렴된 해에 대한 집단 전체의 탐색을 실시하는 유전알고리즘을 결합하고, 보다 정밀한 탐색을 위해서 수렴된 해 주변에 대한 포복법(*creeping method*)을 사용한다.

HGA에서 적용하게 될 지역적 수렴기법과 정밀탐색기법은 다음과 같은 특징을 가지고 있다.

3.1 지역적 수렴(local convergence) 기법

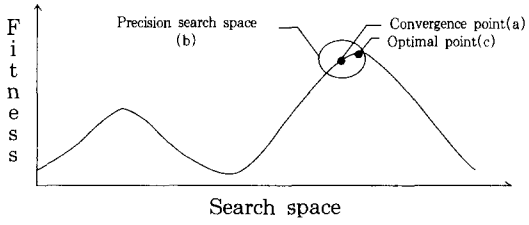
혼합접근법으로서의 지역적 수렴기법은 새롭게 발생되는 개체들을 유전알고리즘의 집단속으로 투입하기 이전에 지역적 최적점으로 이동시키는 것이다. 따라서 이러한 지역적 수렴기법이 각 개체 주위의 지역적 탐색을 위해 사용되어진다면 유전알고리즘은 이렇게 수렴된 집단 가운데에서 전역적 탐색을 수행하기 위해 사용되어질 수 있다. (Davis, 1987)

본 연구에서 사용할 지역적 수렴기법은 랜덤탐색·검사법으로 지역적 수렴점으로 수렴하는 과정에서 생성되는 우수한 개체들을 선택하여 집단으로 만든다. 이러한 선택전략은 확률적 토너먼트 선택전략(stochastic tornament selection strategy)을 변형한 것이다.(Gen and Cheng, 1996)

3.2 정밀탐색(precision search)법

정밀탐색법은 기존의 지역적 탐색과 유전알고리즘을 적용한 후에 사용하는 방법으로 수렴된 해 주위를 정밀하게 탐색하여 더 우수한 해로 이동시키는 기법이다. 이 기법은 기존의 유전알고리즘으로는 그 주위를 정밀하게 탐색할 방법이 없기 때문에 이를 HGA에 적용할 경우 유전알고리즘과 기존의 최적화 기법이 각각 개별적으로 적용될 때 보다 더 우수한 해를 찾을 수

있다.



〈그림 1〉 정밀 탐색 공간

예를들면 〈그림 1〉에서 처럼 만일 유전알고리즘의 수행이후 수렴된 해(a)가 최적점(c) 부근이라면 이 부근에 대해 그 주위(b)의 위·아래를 정밀하게 탐색하여 최적점를 찾을 수 있다는 원리이다.

3.3 HGA의 절차

HGA의 절차는 다음과 같은 8단계로 나누어진다.

먼저 단계 1에서 단계 4까지는 지역적 탐색법인 랜덤탐색·검사법을 적용하여 지역적 최적점으로 빠르게 수렴하도록 하며, 단계 5에서 단계 6까지는 유전알고리즘의 적용단계로 랜덤탐색·검사법에서 생성된 개체를 집단으로 만들어 최적해가 발견될 가능공간(feasible space)에 대한 전역적 탐색을 실시한다. 마지막 단계 7에서 단계 8까지는 유전알고리즘의 적용 이후 수렴된 해 주변의 정밀탐색을 실시하는 포복법을 적용하여 최적해에 도달하도록 한다.

HGA의 수행 종료조건은 미리정의된 반복수만큼 실행한후 종료하는 방법을 사용한다.

단계 1 : 각 설계변수에 대해 미리 정의된 집단의 크기만큼 랜덤하게 각 개체들을 생성하고 이를 제약조건에 대입한후 제약조건을 만족하면 다음단계로 이동하고, 그렇지 않으면 탈락시킨다.

단계 2 : 단계 1을 만족한 개체들을 목적함수에 대입하여 가장 적합도가 높은 개체 하나만을 유전알고리즘 적용을 위한 집단속으로 삽입시킨다.

단계 3 : 단계 1과 2의 과정을 다시 수행하여 이전에 삽입된 개체보다 더 우수한 개체가 생성되면 이를 두 번째의 개체로서 유전알고리즘 적용을 위한 새로운 집단으로 삽입한다.

단계 4 : 단계 1에서 3까지의 과정을 반복수만큼 실행한다.

단계 5 : 단계 1~4에서 생성된 집단을 이용해 유전알고리즘을 실행한다.

먼저 복제(reproduction)단계에서는 룰렛휠(roulette wheel)기법(Goldberg, 1989)을 사용하며, 교차변이(crossover)단계에서는 불일치수리교차변이법(non-conform arithmetic crossover method)(Gen and Cheng, 1996)을, 돌연변이(mutation)단계에서는 랜덤발생돌연변이(random mutation)(Gen and Cheng, 1996)법을 적용한다.

이와같은 유전연산자를 적용해 생성된 개체들에 대해 제약조건에 대입하여 이를 위반하면 탈락시키고 만족하면 목적함수에 대입하여 가장 우수한 개체 하나만을 저장시킨다.

단계 6 : 단계 5를 반복하여 생성된 새로운 우수개체와 이전에 저장된 우수개체를 비교한후 가장 우수한 개체 하나만을 저장시킨다. 이러한 과정을 반복수 만큼 실행시킨다.

단계 7 :포복법에 의한 정밀탐색

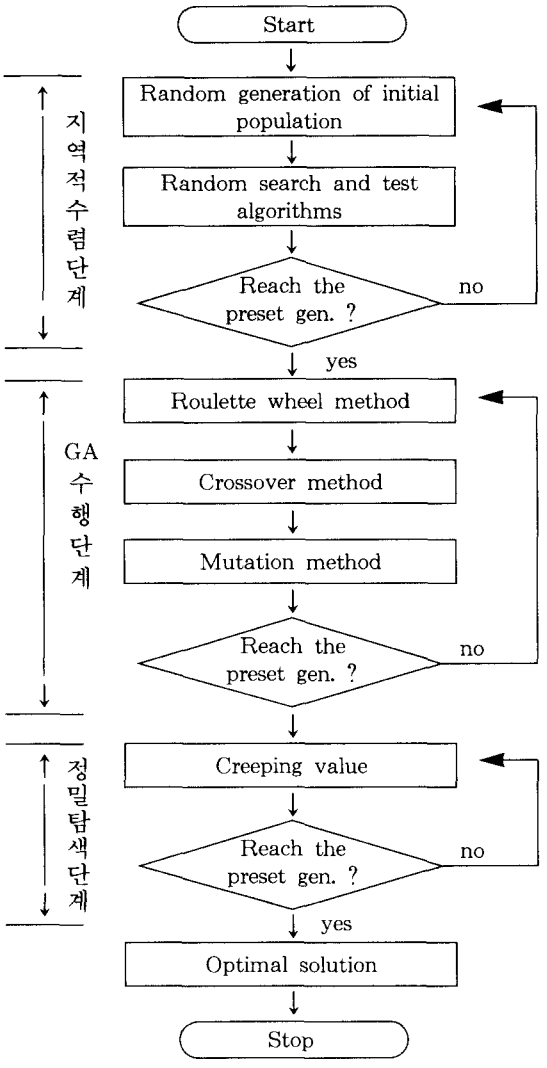
단계 5, 6의 과정을 거치면서 생성된 하나의 우수 개체와 단계 1~4의 과정에서 생성된 하나의 우수개체를 비교하여 가장 우수한 개체 하나만을 이용해 포복법을 실시한다. 이러한 선택전략을 엘리티스트 선택전략(elitist selection strategy)(Gen and Cheng, 1996)이라 하며 개체의 유전과정에서 우수개체의 탈락현상을 막을 수 있다는 장점이 있다. 예를들어 설계변수 x1의 값이 12.5이고 설계자가 정해준 포복값(creeping value)이 1.5라면 x1은 11.0~14.0 범위에 대해 미리 정의된 반복수만큼 정밀하게 재탐색을 실시한다. 이에 대한 변수설계는 다음과 같다.

$$
\begin{aligned}
&for\ i\ =\ 1\ \ to\ \ pop\_size\\
&\quad temp\ =\ (2\times creeping\_value\times rnd)\\
&\qquad\qquad -\ creeping\_value\\
&\quad x_1(i)=x_1\_value+temp\\
&next\ i
\end{aligned}
$$

여기에서 pop_size는 집단의 크기이며, $temp$는 임시기억장소, rnd는 0, 1사이의 랜덤 발생값이고, x_1_value는 HGA수행에서 구해진 지역적 최적 x_1의 값이다. 이렇게 해서 생성된 개체들에 대해 제약조건을 위반하면 탈락시키고 만족하면 목적함수에 대입하여 적합도가 가장 우수한 개체 하나만을 저장시킨다.

단계 8 : 단계 6과 같이 단계 7의 과정을 반복하여 가장 우수한 개체 하나만을 저장시킨다.

본 논문에서 제시한 HGA의 수행과정은 〈그림 2〉와 같다.



〈그림 2〉 HGA의 수행과정

4. 사례연구

본 논문에서 제시하고자 하는 사례모델로서는 기존의 관련 연구문헌에서 사용되는 일반적인 비선형최적화 문제와 공학설계에서 많이 사용되

어지고 있는 기어트레인(gear train)문제와 코일 스프링(coil spring) 최적 설계문제를 이용하였으며 그 결과를 본 논문의 HGA와 비교·분석하고자한다.

4.1 사례모델 1

$$Minimize\ F(x) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$$

Subject to

$$0 \le 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 - 0.0022053x_3x_5 \le 92$$

$$20 \le 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 \le 25$$

$$90 \le 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 \le 110$$

$$78 \le x_1 \le 102$$

$$33 \le x_2 \le 45$$

$$27 \le x_3 \le 45$$

$$27 \le x_4 \le 45$$

$$27 \le x_5 \le 45$$

이 모델은 5개의 독립변수와 3개의 부등제약식을가진 비선형최소화모델로 Himmelblau(1972)가 제안한 후 여러 연구자들에 의해 다양한 기법이 적용되었다. 그 적용결과는 〈표 1〉과 같다.

〈표 1〉 모델 1에의 적용 결과

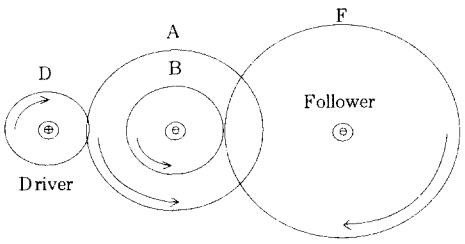
Method	GRG	GA	HGA	Type of variables
x_1	78.620	81.490	78.003	Continuous
x_2	33.440	34.090	33.034	〃
x_3	31.070	31.240	27.108	〃
x_4	44.180	42.200	44.915	〃
x_5	34.370	34.370	44.931	〃
$F(x)$	-30373.950	-30182.296	-31017.086	

〈표 1〉을 살펴보면 Gabriete(1980)는 기존의 최적화 기법인 GRG(generalized reduced gradient method)을 사용하였으며 Homaifar등(1994)은 유전알고리즘(개체의 길이(str_no):19, 집단의 크기(pop_size):400, 교차변이율(p_c):0.8, 돌연변이율(p_m):0.088)을 이용하여 해를 구하였지만, 본 논문의 HGA는 지역적 수렴기법과 수렴해 부근의 정밀탐색을 포함한 유전알고리즘(pop_size): 25, p_c:0.42 p_m:0.4, 포복값($creep\_value$): 1.2, 반복수(gen_no):7500, 실행시간($run\ time$): 33 sec.)으로 기존의 연구들 보다 더 우수한 해를 구하였다. 이러한 결과는 기존의 최적화 기법들이 탐색초기에 한점에서 출발하여 탐색을 실시하기 때문에 해를 다양하게 발생시키지 못하고 지역적 수렴되어 더 이상의 개선이 없으며, 유전알고리즘의 경우도 비록 한점이 아닌 집단으로 탐색을 실시하지만 초기 집단을 랜덤하게 생성하기 때문에 지역적 수렴되어 기존의 지역적 최적화 기법보다도 더 나쁜 결과가 도출되었다.

즉 Homaifar등이 적용한 유전알고리즘의 경우 설계변수의 비트표현을 위해 정밀도를 주게 되지만 설계자가 정밀도를 어떻게 주며, 또한 초기집단이 어떻게 형성되느냐에 따라 그 수렴속도와 구한 해가 상당히 차이가 나게 된다.

4.2 사례모델 2

공학설계에서 많이 사용되는 기어트레인(gear train) 모델은 Sandgren(1990)이 제시했으며 〈그림 3〉에 나타나 있다. 여기에서 사용되는 변수 T_a, T_b, T_d, T_f, 는 각각 A, B, D, F의 톱니의 수이며, 설계규정상 이러한 변수들은 12개에서 60개 사이의 정수값을 가지도록 규정하고 있다. 편의상 T_d, T_b, T_a, T_f를 x_1, x_2, x_3, x_4로 설계변수로 변환하여 사용한다.



〈그림 3〉 기어 트레인의 구조

이에 대한 수리적 모델은 아래와 같다.

$$Minimize\ F(x) = \left(\frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4}\right)^2$$

$$Subject\ to \quad G_i = x_i \geq 12 \qquad i = 1,\ldots,4$$

$$G_i = x_i \geq 60 \qquad i = 1,\ldots,4$$

이상의 모델을 HGA에 적용(*pop_size*: 40, p_c :0.2, p_m :0.1, *creeping_value*:0.5, *gen_no*:30000, *run time*:5 sec.)하여 기존의 최적화 기법과 유전알고리즘을 이용한 Wu등(1995)의 연구와 비교·분석하면 〈표 2〉와 같다.

〈표 2〉 모델 2에의 적용 결과

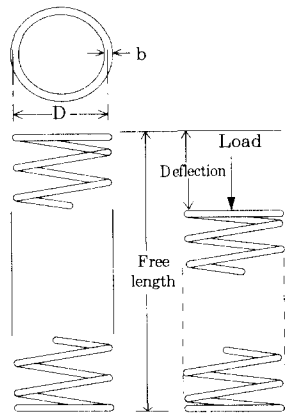
Method	NIDP	Meta-GA	HGA	Type of variables
x_1	18	19	19	Integer
x_2	22	16	16	〃
x_3	45	43	43	〃
x_4	60	49	49	〃
$F(x)$	5.7×10^{-6}	2.7×10^{-12}	2.7×10^{-12}	

〈표 2〉를 분석해 보면 기존의 비선형 이산 정수계획법(nonlinear integer discrete programming :NIDP)을 이용한 Sandgren(1990)의 연구결과보다 Meta-GA를 이용한 Wu등(1995)의 연구와 본 연구의 HGA가 더 우수한 해를 구할 수 있었다.

Sandgran이 제시한 NIDP방법은 일반적인 비선형계획법의 해결절차에 근거를 두고 있으며 문제해결과정에서 초기입력변수값을 어떻게 설정하느냐에 따라 해의 수렴정도가 차이가 난다는 단점이 있고, Wu등이 제시한 Meta-GA는 일반적인 유전알고리즘에서 사용되는 집단의 크기, 교차변이법, 교차변이율, 돌연변이율등을 유전알고리즘의 모수로 사용하여 해결한 것으로 그 과정이 상당히 복잡하다는 단점이 있다. 하지만 본 논문의 HGA는 Meta-GA와 같이 복잡한 탐색과정을 사용하지 않고 지역적 탐색법이 지닌 지역적 수렴특징과 수렴된 해 주위의 정밀탐색을 첨가한 알고리즘으로 Wu등의 연구결과와 같은 해를 도출하였기 때문에 더 간편하고 효율적인 기법이라고 할 수 있다.

4.3 사례모델 3

코일 스프링(coil spring) 최적설계모델은 기계구조의 설계에서 중요한 부품으로 Liebman등(1981)이 제시했으며 구체적인 구조는 〈그림 4〉에 제시되어 있다. 이 문제의 목적함수는 스프링 제작에 사용되는 와이어(wire)의 양을 최소화하는 것으로 여기에는 세 가지의 설계변수가 사용된다. 즉 코일 스프링의 수(N)는 정수변수이며, 굴곡지름(widing diameter: D)은 연속변수이고 와이어의 지름(d)은 미리 규정된 이산설계변수값(Liebman, 1981)이다.



〈그림 4〉 코일스프링의 구조

계산의 편의상 각 설계변수를 다음과 같이 나타낸다.

$$Find\ X = [N, D, d] = [x_1, x_2, x_3]$$

이에 대한 수리적 모델은 다음과 같다.

$$Minimize\quad F(x) = \pi^2 x_2 x_3^2 (x_1+2)/4$$

Subject to

$$G_1(x) = \frac{8 C_f F_{\max} x_2}{\pi x_3^3} - S \le 0$$

$$G_2(x) = l_f - l_{\max} \le 0$$

$$G_3(x) = d_{\min} - X_3 \le 0$$

$$G_4(x) = x_2 + x_3 - D_{\min} \le 0$$

$$G_5(x) = 3.0 - \frac{x_2}{x_3} \le 0$$

$$G_6(x) = \delta_p - \delta_{pm} \le 0$$

$$G_7(x) = \delta_p + \frac{F_{\max} - F_p}{K} + 1.05(x_1+2)x_3 - l_f \le 0$$

$$G_8(x) = \delta_w - \frac{F_{\max} - F_p}{K} \le 0$$

where

$$C_f = \frac{4(x_2/x_3)-1}{4(x_2/x_3)-4} - \frac{0.165 x_3}{x_2}$$

$$l_f = \frac{F_{\max}}{K} + 1.05(x_1 + 2)x_3$$

$$K = \frac{G x_3^4}{8 x_1 x_2^3}$$

$$\delta_p = \frac{F_p}{K}$$

$F_{\max} = 1000\ lb.$

$l_{\max} = 14.0\ inch$

$D_{\max} = 3.0\ inch$

$\delta_{pm} = 6.0\ inch$

$G = 11.5 \times 10^6\ psi$

$S = 189.000\ psi$

$d_{\min} = 0.2\ inch$

$F_p = 300.0\ lb$

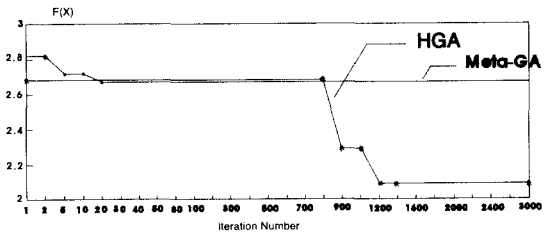
$\delta_w = 1.25\ inch$

이상과 같은 조합최적화 문제에 HGA를 적용(*pop__size*: 20, p_c: 0.2, p_m: 0.1, *creeping__value*: 0.1, *gen__no*: 10000, *run time*: 6.7 sec.)하여 실행한 후 기존의 연구들과 비교·분석한 결과는 〈표 3〉에 제시되었다.

〈표 3〉 모델 3에의 적용결과

Method	Branch & Bound	Meta-GA	HGA	Type of variables
x_1	10	9	9	Integer
x_2	1.180701	1.227411	1.1092	Continuous
x_3	0.283	0.283	0.263	discrete
$F(x)$	2.7995	2.6681	2.0823	

〈표 3〉을 분석해 보면 Sandgren(1990)은 분지한계법(branch and bound)을 이용하였는데 이 기법은 초기출발점을 어떻게 선정하느냐에 따라 탐색시간이 많이 소모되며 설계변수가 혼합되어있는 최적화문제의 경우 설계변수를 정확하게 구현하지 못하고 지역적 수렴되었으며, 기존 유전알고리즘을 이용한 Wu등(1995)의 연구는Meta-GA기법을 이용하였는데 이 기법 역시 이진비트 적용상의 한계로 인해 우수해로의 탐색에 문제점이 나타났다. 즉, 연속변수의 탐색범위를 선정할 때 적용하는 정밀도를 설계자의 경험에 의해 임의로 정하기 때문에 이를 잘못 설계하여 조기 수렴한 것으로 분석된다. 그러나 HGA는 설계변수 자체에 어떠한 변형도 가하지 않고 그대로 사용하기 때문에 더 우수한 해를 구할 수 있었다고 분석된다. 또한 그 구체적인 탐색과정을 살펴보기 위해 Meta-GA와 HGA의 해의 수렴과정을 비교해 보면 〈그림 5〉와 같다.



〈그림 5〉 HGA와 Meta-Ga의 수렴과정

〈그림 5〉를 살펴보면 Meta-GA의 경우 수렴초기단계에서는 HGA보다 우수하지만 반복수가 증가함에 따라 해의 개선이 거의 없는 것을 볼 수 있다. 하지만 HGA는 반복수가 증가함에 따라 해의 수렴이 계속 진행되며 반복수가 22 이상에서는 Meta-GA보다 더 우수한 수렴현상을 나타내고 있다. 이러한 현상은 유전알고리즘만을 적용했을 경우보다 지역적 탐색기법과 정밀탐색을 혼합한 HGA가 더 효율적으로 탐색을 할 수 있는 것으로 분석된다.

5. 결 론

본 연구에서는 지역적 탐색기법인 랜덤탐색·검사법과 수렴된 해 주변을 정밀하게 탐색하는 포복법을 혼합한 유전알고리즘(HGA)을 개발했으며, 이를 일반적인 비선형 최적화 문제와 공학설계에서 많이 사용되는 기어 트레인 및 코일스프링 최적설계모델에 적용하여 HGA의 우수성을 입증하였다. HGA는 기존의 유전알고리즘 적용시 문제가 되었던 해의 지역적 수렴현상, 벌금함수법 및 정밀도의 적용등을 개선시킨 알고리즘으로 다양한 비선형 최적화문제에 효율적으로 적용가능하리라고 본다.

하지만 HGA 역시 설계자가 경험적으로 정해주는 반복수, 집단의 크기, 교차변이율, 돌연변이율, 포복값 등의 범위를 어느 정도로 하느냐에 따라 개체의 다양성 및 최적해로의 수렴속도 등에 많은 영향을 주게 된다. 이러한 현상은 적용하고자 하는 사례문제에 따라 다르게 적용되기 때문에 프로그램 설계시 설계자의 세심한 주의가 필요하다 하겠다.

참고문헌

1) Ackley, D., A Connectionist Machine for Genetic Hill Climbing, Kluwer Academic Publisher, Boston, 1987.

2) Davis, L., Genetic Algorithms and Simulated Annealing, Morgan Kaufman Publishers, Los Altos, 1987.

3) Gabriete, D. and Ragsdell, K., "Large Scale Nonlinear Programming Using the Generalized Reduced Gradient Method," *ASME Journal of Mechanical Design*, Vol. 102, pp. 566-573, 1980.

4) Gen, M. and Cheng, R., Genetic Algorithms and Engineering Design, John Wiley & Sons, pp. 28-30, 1996.

5) Goldberg, D., Genetic Algorithms in Search, Optimizatioon & Machine Learning, Addison-Wesley, 1989.

6) Gorges-Schleuter, M., "ASPARAGOS: An Asynchronous Parallel Genetic Optimization Strategy," *Proceedings of the 3th International Conference on Genetic Algorithms*, San Mateo, California, pp. 422-427, 1990.

7) Gupta, M., Gupta, Y., and Kumar, A., "Minimizing Flow Time Variance in aSingle Machine System Using Genetic Algorithm,"

European Journal of Operational Research, Vol. 70, pp. 289-303, 1993.

8) Himmelblau, M., Applied Nonlinear Programming, McGraw-Hill, Inc., New York, 1972

9) Homaifer, A., Qi, C., and Lai, S., "Constrained Optimization Via Genetic Algorithms," Simulation, Vol. 62, No. 4, pp. 242-252, 1994.

10) Liebman, J. S., Khachaturian, N., and Chanaratna, V., "Discrete Structural Optimization," *Journal of Structural Division*, ASCE, Vol. 107(ST11), pp. 2177-2197, 1981.

11) Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programming, Second Extended Edition, Spring-Verlag, 1994.

12) Michalewicz, Z. and Janikow, C., "Genetic Algorithms for Numerical Optimization," *Statistics and Computing*, Vol. 1, No. 1, 1991.

13) Michalewicz, Z. and Janikow, C., "GENOCOP: A Genetic Algorithms for Numerical Optimization Problems with Constraints," *Communications of ACM*, 1991.

14) Michalewicz, Z. and Janikow, C., "Handling Constraints in Genetic Algorithms," *Proceedings of the 4th International Conference on Genetic Algorithms*, San Diego, July pp. 13-16, 1991.

15) Miller, J., Potter, W., Gandham, R., and Lapena, C., "An Evalution of Local Improvement Operators for Genetic Algorithms," *IEEE Transactions on System, Man, and Cybernetics*, Vol. 33, No. 5, pp. 1340-1351, 1993.

16) Muhlenbein, H., How Genetic Algorithms Really Work: Part Ⅰ, Mutation and Hill-Climbing, Elservier Science Publishers, North-Holland, pp. 15-26, 1992.

17) Reeves, C., "Genetic Algorithms and Neighborhood Search," *Evolutionary Computing*, Springer-Verlag, Berlin, pp. 115-130, 1994.

18) Sandgren, E., "Nonlinear Integer and Discrete Programming in Mechanical Design Optimization," *ASME Journal of Mechanical Design*, Vol. 112, No. 2, pp. 223-229, 1990.

19) Wu, S. J. and Chow, P. T., "Genetic Algorithms for Nonlinear Mixed Discrete-Integer Optimization Problems via Meta-Genetic Parameter Optimization," *Engineering Optimization*, Vol. 24, pp. 137-159, 1995.

20) Zheng, D., Gen, M., and Chen, R., "Multiple Objective Optimization Using Genetic Algorithms," *Proceeding of the 20th International Conference on Computers & Industrial Engineeering*, pp. 117-120, 1996.