

System의 수행도를 개선시키기 위한 유전자 알고리즘의 도입에 관한 연구

A Study on the Introduction of Genetic Algorithms for Developments Performance of System

김 병 석* · 김 용 범* · 장 병 집*

B.S. Kim · Y.B. Kim · B.J. Jang

(1998년 6월 3일 접수, 1998년 12월 11일 채택)

ABSTRACT

This paper proposed a method for solving the nonlinear integer programming problem to get easily the best compromise solution while holding a nonlinear property by using the genetic algorithms. Also, this paper reported that the optimization problem of systems reliability as was solved by using the proposed method, and the numerical comparison experiments between the 0-1 LP/0-1 NP formulations were demonstrated, and from the quantitative evaluation the efficiency of the proposed method was demonstrated.

1. 서 론

최근 장비의 복잡성의 증가로 인해 사회 전반적인 산업은 고도의 성능, 신뢰도, 보전도, 안전도 등을 포함하는 새로운 공학적인 문제들이 대두되어지고 있다. 이러한 측면에서 안전도, 보전도, 신뢰도의 결합된 측정의 활용가능성은 시스템의 효능을 측정하는 것으로써 그 효능을 광범위하게 증대시킨다. 이중에서도 다양한 여러

조건하에서 주어진 임무에 대한 시스템 신뢰도의 향상은 산업적으로, 군사적으로, 일상 생활적으로도 매우 중요한 것들 중의 하나이다.

비록 신뢰도에 있어서의 정성적 개념에 대한 새로운 사실이 없을 지라도, 정량적인 측면에 있어서는 과거 수년동안 점진적으로 발전되어지고 있는 실정이다. 이와 같은 발전은 고도의 시스템 신뢰도에 대한 필요성의 증대 및 더욱 높은 안전도와 비용의 손실을 줄이기 위한 방편의

* 충주산업대학교

산물이었다.

시스템의 신뢰도를 개선시키는 데에는 여러 기법들이 사용되어 왔다. 이러한 기법들 중에 몇몇은 거대한 안전인자들을 이용하거나, 시스템의 복잡성을 줄이거나, 개선된 프로그램을 통해 구성된 부품들의 신뢰도를 증가시키거나, 구조적인 중복(redundance)을 이용하거나, 또는 장비의 유지/보수 스케줄을 활용하거나 하는 방법들에 의해 대상문제에 적용한다. 이러한 문제를 해결하는데 있어서 대부분은 최적의 신뢰도를 구하기 위한 노력으로 집중되고 있다⁹⁾.

최적의 신뢰도를 구하기 위한 기법은 다른 기법들에 비해 선형이나 비 선형 계획문제를 해결하는데 있어 그들 고유의 특성과 만족할 만한 우수성을 가지고 있다. 이러한 시스템의 최적 신뢰도를 구하기 위한 기법으로는 정수계획법(integer programming), 동적계획법(dynamic programming), 리그랑지와 쿤터커의 조건을 이용한 기법(method of Lagrange multipliers and Kuhn-Tucker conditions) 등 여러 기법이 이용되어지고 있다. 위와 같은 기법들 중에서 실생활에 이용되는 대상문제의 대부분은 선형 또는 비선형 정수계획문제가 차지하고 있다. 선형 또는 비 선형 정수계획문제는 그 해가 정수이어야만 한다는 가정으로 인해 일반적인 선형 또는 비 선형 계획문제를 보다 해를 도출하기가 어렵다.^{1,12)}

정수계획법(Integer Programming : IP)은 선형인 제약조건과 변수들이 정수이어야 한다는 제약을 갖는 최대화(최소화) 선형목적함수를 만족하는 수리 계획적인 모형이다. 몇 개의 변수만이 실수이어야 한다는 조건을 갖는 정수계획모형을 혼합정수계획법(Mixed Integer Programming : MIP)이라 한다^{2,6)}.

Branch-and-Bound 기법은 선형인 정수계획문제를 해결하는데 가장 광범위하게 이용되어지는 알고리즘이다. 그렇지만, Taha에 의하면 Branch-and-Bound 기법은 근본적으로 branching 규칙은 선형이어야 한다는 가정으로 인해 비선형 정수계획 문제를 해결하는 데에는 적합하지 않다²⁾.

Dakin에 의해 제안된 수정된 Land-Doig 기법

은 선형성의 조건에 상호 의존하는 branching 규칙을 가정한 것이다³⁾. 그러나 이 기법은 결정 변수(decision variable)에 관련된 목적함수와 제약함수가 오목(concave)과 볼록(convex)이라는 가정 하에서만, 국부 최적해(local optimal)가 전체 최적해(global optimal)로 된다는 전제하에 비선형 정수계획법을 풀이할 수가 있다.

비선형 정수계획법을 풀기 위한 또다른 기법으로는 Taha에 의해 제안된 intelligent enumeration 기법이 있다²⁾. 이 기법은 단지 대상문제가 다항(polynomial)인 정수계획문제를 해결하는 것이다. 다항인 정수계획문제를 풀기 위한 또 다른 branch-bound method가 Glover와 Woolsey⁴⁾에 의해 제안되었다. 그러나 이 기법은 근본적으로 비선형 정수계획문제를 해결하는 과정에서 목적함수에 제약조건들의 곱의 형태를 결합시켜야 하는 관계로 인해 계산시간과 저장공간이 극단적으로 증가하는 문제가 있다. 최근에 Li는 비선형 정수계획 문제를 해결하는데 있어 기존의 비선형 계획법에 근거한 근사 방법을 제안하였다⁶⁾. 그러나 이 기법은 대상문제를 풀이하는 과정에 있어서 0-1 이진변수들로 상한(upper bound)과 하한(lower bound)을 갖는 모든 정수 x_j 를 변형시켜야 하는 번거러움이 있다.

최근에 주어진 대상문제의 해를 구하는 기법의 하나로 조합최적화(combinatorial optimization) 문제를 효과적으로 해결하는데 있어 유전자 알고리즘(genetic algorithms)의 적용은 매우 큰 효과를 나타내고 있다^{7,8)}. 유전자 알고리즘은 시스템 신뢰도의 최적 설계와 같은 비선형 정수계획 문제를 해결하는데 매우 강력한 도구로써 이용되어지는 것으로 그 대상문제의 목적함수(objective function)와 제약조건(constraint)들이 어떠한 형태이든 쉽게 접근할 수 있는 기법이다. 따라서 본 연구에서는 유전자 알고리즘을 이용하여 대상문제가 비선형의 가정이 유지되는 비선형 정수계획문제를 해결하는 과정에 있어서 최상의 절충된 해(compromise solution)를 쉽게 얻을 수 있도록 본 연구에서 제안한 기법을 적용하고자 한다. 또한 적용사례로써 시스템 신뢰도의 최적화 문제를 제안한 기법에 의해 해결하고, 0-1 선형계획법/0-1 비선형 계획법과 수치

적 실험을 통해 비교하여 본 연구에서 제안된 기법이 정량적으로 해를 구하는 과정에 있어서 효과적임을 입증하고자 한다.

2. 유전자 알고리즘

유전자 알고리즘은 여러 개의 개체(individual)가 병렬 적으로 동시에 주어진 환경에 따라 적자생존(fittest survival)의 방법으로 진화하여 궁극적으로는 최적의 상태에 도달하는 생태계의 진화이론에서 따온 알고리즘으로 Holland에 의해 처음으로 도입되었다.

유전자 알고리즘에서 여러 개의 개체들로 구성된 세대(generation)가 진화(evolution)할 때 이전의 세대가 특정한 환경에 대한 정보는 염색체(chromosome)의 형태로 표현되어 다음 세대에 전달된다. 이때 조상(parents)의 염색체가 그대로 복제(reproduction)되어 자손(child)에 전달되는 것이 아니라 조상의 염색체에 선택(selection), 교차(crossover), 돌연변이(mutation) 연산자(operator)를 이용한 형태의 염색체로 전달된다. 이러한 과정에서 조상의 세대 중에서 한 개체가 선택되어 자손에 유전정보를 남길 확률은 일반적으로 그 개체의 상호작용에 의존하는 값에 따라 변한다⁷⁾.

유전자 알고리즘은 개별 공정 일정계획(job shop scheduling), 기계학습(machine learning), 시스템 신뢰도, 경로배정문제, 형상인식(pattern recognition) 등과 같은 광범위한 분야에서의 최적화 문제에 적용되어 좋은 결과를 보여주고 있다. 유전자 알고리즘의 구조는 다음 Fig. 1과 같다⁸⁾.

유전자 알고리즘에 의해 일반적으로 이용되어지는 유전연산자로는 선택, 교차, 돌연변이 등이 있다. 이러한 유전연산자들은 생물학적인 진화과정을 통하여 도출된 것으로 다음 세대를 구성하는데 결정적 역할을 한다.

procedure evolution program

```

begin
t ← 0
initialize P(t)
while(not termination-condition) do

```

```

begin
t ← t+1
select P(t) from p(t+1)
alter P(t)
evaluate P(t)
end
end

```

Fig. 1 Structure of genetic algorithms

2.1 교차연산자

교차연산자(crossover operator)는 중요한 유전연산자이다. 이것은 결합된 두 조상의 특성에 의해 하나의 자손(offspring)을 생성시키는 것으로 동시에 두 개체에 적용된다. 교차는 랜덤하게 선정된 개체 s_i 로부터 역시 랜덤하게 선정된 교차점(crossover point) 앞의 유전인자(gene) 부분과 또 다른 개체 s_j 로부터 교차점(crossover point) 뒤의 유전인자 부분을 연결하여 이미 존재하는 개체 s_i 와 s_j 로부터 새로운 개체인 s'_i 와 s'_j 를 생성하는 것이다. 이러한 유전인자의 부분들은 유전인자의 $l-1$ 위치들에서 균등분포(uniform distribution)로부터 랜덤하게 선정된 교차점에 의해 정의된다. 예를 들어, 만약 교차가 3번째와 4번째 유전인자 사이에서 발생되었다면, 개체 s'_i 와 s'_j 는 s_i 와 s_j 로부터 다음과 같이 생성된다¹³⁾.

Parent1: $s_i = a_{i1} a_{i2} a_{i3} | a_{i4} \dots \dots \dots a_{in}$

Parent2: $s_j = a_{j1} a_{j2} a_{j3} | a_{j4} \dots \dots \dots a_{jn}$

↑
교차점(pos)

Offspring1:

$s'_i = a_{i1} a_{i2} a_{i3} a_{j4} \dots \dots \dots a_{in}$

Offspring2:

$s'_j = a_{j1} a_{j2} a_{j3} a_{i4} \dots \dots \dots a_{jn}$

교차연산자는 다음과 같은 방식으로 유전자 알고리즘에 이용한다. 자손을 생성하기 위해 주어진 개체군으로부터 선정된 개체 s_i 이 주어지면, 또 다른 개체 s_j 는 선택확률을 이용하여 개체군으로부터 선정한다. 그러면 새로운 자손은 교차에 의해 생성된다. 따라서 교차는 기본적인

알고리즘의 적합도에 의해 선정된 점들의 샘플로부터 새로운 점들의 샘플인 더 좋은 해들을 생성하기 위해 해 공간의 탐색에 적용된 전략이다.

예를 들어, 본 연구에서 교차점 $pos=3$ 이라 하면 다음과 같이 염색체의 교차가 이루어진다.

Parent 1: $s_i = 2\ 5\ 0\ | 1\ 2\ 3\ 2\ 1$

↓ ↑

Parent 2: $s_j = 1\ 0\ 0\ | 4\ 5\ 3\ 2\ 2$

Offspring 1: $s'_i = 2\ 5\ 0\ 4\ 5\ 3\ 2\ 2$

Offspring 2: $s'_j = 1\ 0\ 0\ 1\ 2\ 3\ 2\ 1$

교차비율(crossover rate)은 개체군의 크기에 대한 각 세대에서 생성된 자손 수의 비율로 정의된다⁷⁾.

2.2 돌연변이 연산자

일반적으로 자연계에서 한 유전인자에 돌연변이가 진행될 확률은 0.001보다 적은 값으로 적용한다⁷⁾. 이것은 어떠한 유전인자도 개체군으로부터 영구히 소멸되지 않는다는 것을 보장하는 배경 연산자(background operator)로서 적용된다. 돌연변이를 수행하기 위한 가장 단순한 방법은 한 쌍의 유전인자를 상호 교환하는 것이다. 유전자 알고리즘에서 돌연변이는 초기 개체군에서 표현되지 못했던 유전인자들을 제공하거나, 다음 세대를 구성하는 선택과정이 진행되는 동안 개체군으로부터 손실되었던 유전인자들을 복구하는 결정적 역할을 제공한다.

예를 들어, 랜덤 하게 선정된 염색체에 돌연변이($j=5$, $d_b=[0, 6]$)가 발생되었다면 다음과 같다.

$s_i = 3\ 2\ 0\ 2\ 2\ 1\ 4\ 5$

↓

$s'_i = 3\ 2\ 0\ 2\ 4\ 1\ 4\ 5$

돌연변이 비율은 개체군에 있어서 모든 유전인자들에 대한 퍼센티지로 정의된다. 또한 이것은 알고리즘이 수행되는 동안 유전인자들이 개체군내로 새롭게 생성되는 비율을 조정한다.

2.3 선택연산자

유전자 알고리즘은 exploitation과 exploration간의 대립에 의해 수행된다. 여기서 exploitation은 해의 탐색과정에서 이미 얻어진 정보를 적절하게 활용한다는 장점이 있다. 예를 들면, 개체군에서 유전인자 1010을 갖는 string들이 보다 개선된 적합도 값을 갖는 string들이라고 가정하자. 만약 세대가 진행되는 과정에서 1010의 형태를 갖는 이러한 유전인자들의 위치들이 빠르게 고정(fixing)된다면, exploitation은 개체군의 평균 적합도를 개선하는데 이러한 정보를 이용하게 된다. 그러나 개체군이 너무나 빠르게 이러한 유전인자들로 구성된다면 사전수렴(pre-mature)이라는 위험이 따르게 된다. 여기서 이와 같은 사전수렴을 방지해주는 것이 즉, string에서 유전인자들의 또 다른 형태가 더 효과적일 수 있다는 것을 보여주는 것이 exploration이다. 유전자 알고리즘에서는 exploitation과 exploration간의 이와 같은 경향이 항상 존재하여 세대가 진행되면서 더욱 개선된 해를 탐색하는 기법이다. 이와 같은 exploitation과 exploration간의 적절한 균형을 맞추기 위한 결정적 역할을 하는 것이 선택기법이다¹³⁾. 기존의 유전자 알고리즘에서 이용된 선택기법은 여러 가지가 있으나, 본 연구에서는 염색체의 가장 좋은 적합도(fitness) 값을 갖는 유전인자가 계속해서 다음 세대를 구성하는 ranking 선택기법을 이용하여 자손을 생성시키고자 한다¹³⁾.

최근 수년동안 유전자 알고리즘은 기존의 여러 기법들에 대한 새로운 도전의 결과로서 빠르게 변화되어 왔다. 이러한 유전자 알고리즘의 변화 중에서 가장 탁월한 기법들 중의 하나는 진화연산(evolution program)이다⁸⁾. 진화연산은 전적으로 유전자 알고리즘에 기반을 둔 것으로, 유전자 알고리즘이 고정된(fix) 길이를 갖는 이진수(binary)를 이용하는 것에 비해 진화연산은 주어진 대상문제에 대해 실제 데이터의 구조로 잠재적인(potential) 해를 자연적인(natural) 방식으로 표현하여 아주 유용한 근사해(approximation of solution)를 찾는 것이다.

3. 알고리즘의 개발

연구에서는 다음과 같이 비선형 정수계획문제를 풀기 위해 유전자 알고리즘에 근거를 둔 기법을 개발하고자 한다.

단계 1: 랜덤 하게 n 개의 원소를 갖는 초기 염색체 $s_k(k=1, \dots, pop_s)$ 를 다음과 같이 생성한다. 여기서 pop_s 는 개체군의 크기이다.

$$s_k = [x_{k_1}, x_{k_2}, \dots, x_{k_n}]$$

염색체를 생성할 때 다음과 같은 조건을 만족해야 한다.

- 1) 각 염색체의 원소들의 값의 범위는 다음과 같은 범위를 만족해야 한다.

$$x_{kj}^L \leq x_j \leq x_{kj}^U$$

- 2) 각 염색체는 다음과 같이 모든 제약조건(constraint)들을 만족해야 한다.

$$g_i(s_k) = b_i, \quad \forall i$$

한 개의 염색체라도 위의 조건들을 만족시키지 못한다면, 대상문제는 가능해(feasible solution)를 찾을 수 없고 계산을 끝낸다.

단계 2: $gne = gne + 1$ 로 놓고, 평가함수(evaluation function) $eval(s_k)$ 를 계산한다. 여기서 gne 는 세대수(number of generation)이다.

$$eval(s_k) = f(s_k), \quad k = 1, 2, \dots, pop_s.$$

단계 3: 교차: 교차에 의해 생성된 염색체의 수를 $ccnt = 0$ 이라 하자. 범위 $[0, 1]$ 에서 난수 $r_k(k=1, 2, \dots, pop_s)$ 를 생성한다. $r_k < p_c$ 를 만족하는 s_k 를 선택한다. 여기서 p_c 는 교차확률이다. s_k 중에서 한 쌍을 만들고, $ccnt = ccnt + 2$ 로 놓는다. 랜덤하게 교차점(crossover point)을 선정하고, 교차를 수행한다. 교차의 과정에서 새롭게 생성된 염색체를 s'_{ccnt-1} 과 s'_{ccnt} 라 한다.

돌연변이: 돌연변이에 의해 생성된 염색체의 수를 $mcnt = 0$ 이라 하자. 범위 $[0, 1]$ 에서 $r_k(k=1, 2, \dots, pop_s + n * ccnt)$ 를 생성한다. $r_k < p_m$ 을 만족하는 유전인자(gene)를 선택한다. 여기서 p_m 는 돌연변이확률이다. $mcnt = mcnt + 1$ 로 놓는다. 같은 염색체 내의 돌연변이의 경우에 있어서 $mcnt$ 의 값을 증가시킬 수는 없다. 돌연변이 x_{kj} 는 다음과 같은 범위 내에서 선정한다.

$$x_{kj}^L \leq x_{kj} \leq x_{kj}^U$$

돌연변이의 수행과정에서 새롭게 생성된 염색체를 $s'_{ccnt+mcnt}$ 라 한다.

선택: 새롭게 생성된 평가함수 $eval(s'_t), t=1, 2, \dots, ccnt+mcnt$ 를 계산한다. 시스템의 제약조건을 만족하지 않는 염색체에 대해서는 벌점(penalty)을 부과한다. 보다 좋은 값을 갖는 조상 $s_k(k=1, 2, \dots, pop_s)$ 와 새롭게 생성된 자손 $s'_t(t=1, 2, \dots, ccnt+mcnt)$ 사이에서 염색체들을 선정한다. 같은 구조를 갖는 염색체들을 선정하지 않고 다음 세대를 구성하는 염색체를 s_k 라 한다.

단계 4: 만약 $\max eval < \max \{eval(s_k)\}$ 라면,

$$s^* = \arg \max \{eval(s_k)\}$$

$$\max eval = \max \{eval(s_k)\}$$

여기서 $\max eval$ 은 평가함수의 값이다.

단계 5: 만약 $gne < gne_{max}$ 이면, 단계 2로 간다. 만약 $gne = gne_{max}$ 이면, s^* 를 출력하고 알고리즘을 종결한다.

4. 실험적 연구

본 연구에서는 비선형 제약조건을 갖는 시스템에 대한 최적 신뢰도의 설계에 대해 실험회수를 30회 제한하여 그중 적어도 28회가 전체 최적해(global optimum)에 수렴할 때 시행이 성공적이라는 가정하에 실험에 임하였다. 비선형 제약조건을 갖는 시스템의 최적 신뢰도 설계 시 심지어 여러 원소들 중에서 하나의 고장(failure) 모드(mode)가 발생하더라도 최적 신뢰도의 설계가 어려워진다. 따라서 이러한 문제를 해결하는 한 방편으로 추계적 오차(stochastic error)를 줄일 수 있게 교차확률 $p_c = 0.4, 0.6, 0.8$ 로, 돌연변이 확률 $p_m = 0.01$ 로, 개체군의 크기 $pop_s = 50, 100$ 로 하여 알고리즘을 수행하였다. 본 연구의 실험에는 개인용 퍼스널 컴퓨터(586 Pentium 166MHz)와 Mathematica Ver. 2.2.3을 이용, 정량적인 값을 구하기 위한 제안된 기법의 효과와 불완전한 FDS(Fault Detecting & Switching)를 고안하기 위한 최적신뢰도에 대한 수치적 실험을 수행하였다^{10,11)}. 본 실험의 대상문제로는 다음과 같은 비선형 정수계획문제를 대상으로 하였다.

$$\max f(x)$$

$$\begin{aligned} \text{s.t } & g_i(x) \leq 0, \quad i = 1, 2, \dots, m \\ & x_j^L \leq x_j \leq x_j^U, \quad j = 1, 2, \dots, n \\ & x_j \geq 0, \text{ and integer}, \quad j = 1, 2, \dots, n \end{aligned}$$

여기서 $f(x)$ 와 $g_i(x)$, $i=1, 2, \dots, m$ 는 비선형인 실수 값을 갖는 함수이고, x_j^L 과 x_j^U 는 각각 x_j 의 하한(lower bound)과 상한(upper bound)이다.^{10,12)}

비선형 정수계획문제는 또한 다음과 같이 4개의 부분 시스템에 대한 중복적(redundant)구조를 갖는 최적 신뢰도 문제로 모델화 하였다. 이러한 비선형 목적함수와 제약조건에 대한 모든 상수들은 Table 1에 제시하였다.

Table 1 Data of nonlinear integer programming problem

i	d_{1i}	d_{2i}	d_{3i}	β_i	p_i	q_i
1	1	7	7	0.2	0.93	0.07
2	2	7	8	0.06	0.92	0.08
3	3	5	8	0.0	0.94	0.06
4	4	7	6	0.3	0.91	0.09

$$\max R(x) = \prod_{i=1}^4 R_i(x_i)$$

$$\text{s.t } G_1(x) = \sum_{i=1}^4 d_{1i}(x_i)^2 \leq b_1$$

$$G_2(x) = \sum_{i=1}^4 d_{2i} \left(x_i + \exp\left(\frac{x_i}{4}\right) \right) \leq b_2$$

$$G_3(x) = \sum_{i=1}^4 d_{3i} \left(x_i + \exp\left(\frac{x_i}{4}\right) \right) \leq b_3$$

$$[x_j^L, x_j^U] = [1, 6], \quad j = 1, 2, 4$$

$$[x_3^L, x_3^U] = [1, 5]$$

여기서 $R_i(x_i)$ 는 다음과 같다.

$$R_1(x_1) = 1 - q_1((1 - \beta_1)q_1 + \beta_1)^{x_1 - 1}$$

$$R_2(x_2) = 1 - \frac{\beta_2 q_2 + p_2 q_2^2 (1 - \beta_2)^{x_2}}{p_2 + \beta_2 q_2}$$

$$R_3(x_3) = 1 - (q_3)^{x_3}$$

$$R_4(x_4) = 1 - q_4((1 - \beta_4)q_4 + \beta_4)^{x_4 - 1}$$

Table 1은 불완전한 FDS를 개선하기 위해 대상문제의 최적신뢰도를 모델화한 비선형 정수 계획문제의 목적함수와 제약조건의 모든 상수들

을 제시한 것이다. 또한 알고리즘을 수행한 후 결과는 다음과 같고, Fig. 2는 그 결과를 도시한 것이다.

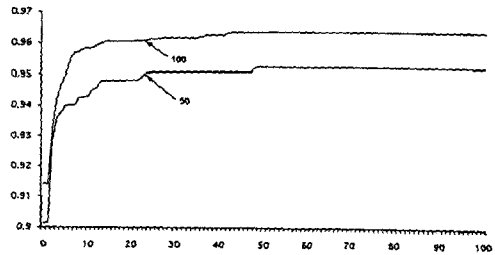


Fig. 2 Convergence of genetic algorithms for population size 50, 100

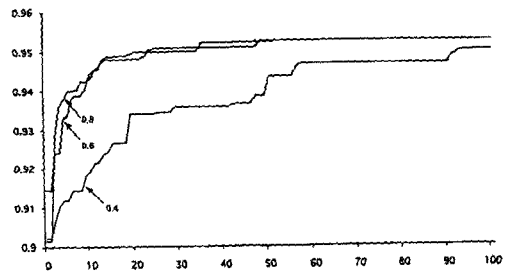


Fig. 3 Convergence of genetic algorithms for crossover probability 0.4, 0.6, 0.8

best chromosome

gen : 48

v* : [3 3 2 3]

eval(v*) : 0.975545

x1 : 3

x2 : 3

x3 : 2

x4 : 3

objective function

Z : 0.975545

row information of system constraints

	G(x)	SP	RHS
G1	75.000	25.000	100.000
G2	125.7006	24.2994	150.000
G3	159.7505	0.2495	160.000

이와 같은 비 선형 정수계획문제에서 각 부분 시스템의 최적 중복(redundant) 단위(unit)들은 $x_1=3, x_2=3, x_3=2, x_4=3$ 이고, 이때 시스템의 신뢰도는 0.975545이다. 또한, 시뮬레이션 수행 시 추계적 오차를 줄이기 위해서 교차확률 $p_c=0.8, 0.6, 0.4$ 로 두고 대상문제에 가장 적합한(adaptive) 모수(parameter)를 구하였으며, 그 결과를 Fig. 3에 도시하였다. 이와 같은 수치적 실험의 결과로부터, 개체군이 매우 클 때 적합도 평가함수는 안정적으로 수렴한다는 것을 알 수 있다.

알고리즘 수행 시 개체군의 크기에 따른 컴퓨터 계산시간은 약간의 차이는 발생하지만 평균적으로 약 449초 정도의 시간이 소요된다. 같은 비선형 정수계획문제를 0-1 비선형계획법/0-1 선형계획법으로 풀었을 경우 본 연구에서 제안한 기법과 같은 최적 해를 얻을 때까지 걸리는 평균 계산시간은 각각 약 628초, 739초 정도의 시간이 소요되므로 약 65%정도의 계산시간을 줄일 수 있었고, 이러한 결과를 Table 2에 제시하였다.

Table 2 The comparison results for each method (NIP/0-1 NP/0-1 LP)

기 법	NIP	0-1 NP	0-1 LP
결정변수의 수	n (4)	$\sum_{i=1}^n L_i$ (12)	$\sum_{i=1}^n u_i$ (23)
제약조건의 수	T (3)	T (3)	T+S (7)
계산시간(초)	420.3-480.5	590.5-667.6	698.8-776.7

5. 결 론

본 연구에서는 유전자 알고리즘을 이용하여 시스템의 최적 신뢰도를 구하는데 있어서 가장 적절한 해를 쉽게 얻을 수 있는 비선형 정수계획문제를 풀 수 있는 기법을 제안하였다.

또한 본 연구에서 제안한 기법과 0-1 비선형 계획법, 0-1 선형계획법에 대한 수치적 비교 실험을 통해 시스템 신뢰도의 최적화 문제를 구하였다. 본 연구의 실험은 비 선형 제약조건을 갖는 시스템의 최적 신뢰도의 설계에 대해 실험회수를 30회 제한하여 그중 적어도 28회가 전체 최적해(global optimum)에 수렴할 때 시행이 성

공적이라는 가정하에 실험에 임하였다. 그 결과 시스템의 최적 신뢰도를 구하는 과정에 있어서 본 연구 기법의 도입은 계산 처리 시간이나 저장공간 등에 있어서 효율적임이 입증되었다. 또한 시스템의 최적 신뢰도를 구하는 과정에 있어서 본 연구의 기법을 이용하면, 주어진 대상문제가 비-연속이며, 비-볼록(non-convex), 다봉(multimodal)인 경우에 있어서도 쉽게 구할 수 있다는 장점이 있다.

앞으로 계산의 복잡도를 줄이기 위해 해의 신뢰도에 영향을 주지 않으면서 가능한 적은 개체군의 크기를 선정하는 대신에 개체군들 내의 적절한 Building-Block의 결합을 이용하여 더 많은 분열을 발생시키는 교차연산자를 이용하면, 알고리즘의 계산시간 및 해의 질을 상대적으로 향상시킬 수 있다.

참 고 문 헌

- 1) D.G. Luenberger, "Linear and Nonlinear Programming", 2nd ed, Addison Wesley, 1984.
- 2) H. Taha, "Integer Programming", Academic Press, 1975.
- 3) R. Dakin, "A tree-search algorithm for mixed integer programming problems", Computer J., Vol. 8, pp. 250~255, 1965.
- 4) F. Glover and E. Woolsey, "Converting 0-1 Polynomial programming problems to a 0-1 Linear program", Ops. Res. pp. 180~182, 1974.
- 5) A. Land and A. Doig, "An automatic method for solving discrete programming problems", Econometrics, pp. 497~520, 1960.
- 6) H.L. Li, "An approximate method for local optima for nonlinear mixed integer programming problems", Computers & Ops. Res., Vol. 19, No. 5, pp. 435~444, 1992.
- 7) D.E. Goldberg, "Genetic Algorithms in Search Optimization & Machine Learning", Addition Wesley, 1989.

- 8) Z. Michalewicz, "Genetic Algorithms+Data Structure=Evolution Program", Springer-Verlag, 1992.
 - 9) F. Tillman, C.L. Hwang and W. Kuo, "Optimization of systems reliability", Dekker, 1980.
 - 10) F.A. Tillman, "Optimization by integer programming of constrained reliability problems with several modes of failure", IEEE Trans. on Rel., Vol. R-18, pp. 47~53, 1969.
 - 11) C.L. Hwang, H.B. Lee, F.A. Tillman and C.H. Lie, "Nonlinear integer goal programming applied to optimal system reliability", IEEE Trans. on Rel., Vol. R-33, pp. 431~433, 1984.
 - 12) W. Kuo, H.H. Lin, Z. Xu, W. Zhang, "Reliability optimization with the Lagrange multiplier and Branch and Bound technique", IEEE Trans. on Rel., Vol. R-36, pp. 624~630, 1987.
 - 13) 김용범, "유전자 알고리즘의 수렴현상에 관한 연구", 박사학위논문, 1996.
-